APPENDIX - A

DEALERS GUIDELINES/CRITERIA

WES'ı LAKE FINANCIAL SᴇRVICES

HOW TO STRUCTURE A DEAL USING THE WESTLAKE BUY MODEL

Step 1
Credit Application
- Include nearest relative and landlord information.
- Know income, debts, and time on job.

Step 2
Review Credit Bureau Printout
- Count good and derog items.
- Time on bureau
- See *Credit Bureau Input Guidelines*

Step 3
KELLEY BLUE BOOK
- Wholesale Book + approved adds
- Deduct High Miles
- **Do NOT add for low miles**

Step 4
Deal Structure
- Price, down, term
- Fine-tune deal
- Adjust price, down, term, reserve
- Check Amt Fin
- Complete Stips

SCREEN DEFINITIONS

#Years on Credit Bureau- The earliest record of credit. You may count Collection Accounts from the date assigned, but not inquiries. If no credit, input 0.0.

# Good Credit Items- Count the number of "good" pieces of credit on the bureau, using Westlake guidelines. "Good" items that are later reported as "derog" items do not count. Multiple "good" entries on the same account do not count. Do not count child support (F/S) accounts. Do not count student loans. **NOTE:** There are some accounts that are both "good" and "derog."

$ High Good Credit- The highest credit line **ever established** on an account classified as "good" credit. *Child support (F/S) and student loans do not count.*

# Derog Credit Items- Count the number of "derog" pieces of credit on the bureau, using Westlake guidelines. Accounts charged off with Bankruptcy are derog. Repos count as both 1 "derog" and 1 under "# of Repossessions." Do not count child support (F/S) accounts. Do not count student loans. **Do not count Medical Collection Accounts. NOTE:** There are some accounts that are both "good" and "derog."

$ High Derog Credit- The highest amount **ever established** on any "applied for" account classified as "derog" credit. *Do not count tax liens, student loans, or child support.*

# of Repossessions/Auto Losses- Count all repos, voluntary surrender, redeemed repos, paid repos, charged-off autos, BK LIQ autos, insurance deficiency autos, and any other autos (or installment loans from any lender that makes auto loans that **could be** an auto loan) that appear to have **ever** been repo'd or skipped or resulted in any form of a loss to a creditor. If spouse is on the contract, input total # of repos between them.

Prev Bankruptcy: Y/N- Input Y if BK or if any BK accounts appear on credit bureau, whether buyer or spouse. Must be discharged or dismissed.

Customer Owns Home: Y/N- Must supply documentation to input Y. Must be current on mortgage or be able to prove mortgage is current. Must live in the house they own. Mobile homes not eligible unless customer owns the land as well as the home.

Residence Stability #- Since age 18. Check credit and driver's license for any conflict. Must be able to contact landlord.

#Years on Present Job- Since age 18. Be sure to get **verifiable** info from the customer. Foster Care, Home Care, AFDC, SSI, welfare, and any other type of local, state, or federal assistance input 0.0. If retired or permanently disabled, input 2 years. Self-employed, letter POI, or family business not more than 2 years unless can **prove** otherwise. Temp jobs / agencies input 0.1 years unless we can verify with the **employer,** not the agency. If seasonal employee, use max of 2 years employment.

Gross Monthly Income- Verifiable gross income before taxes. Foster Care, Home Care, AFDC, SSI, welfare and any other type of income received due solely to the existence of another person count 50%. No food stamps, student loans or grants may be counted as income. Soft POI (s) input justifiable income to max $1500/mo. If **paying** child support on bureau or paystub, **deduct the amount from income (Windows Program: Input in "Family Support Debt").** Don't add it to Other Monthly Debts.

Rent/Mortgage- Input Rent or Mortgage Payment.

Other Monthly Debts- All monthly payments listed on application or still active/open on credit bureau, besides rent or mortgage. If no payment shown count 5% the balance. Be sure to count any garnishments or discretionary allotments on paystub if not listed on credit bureau, except for child support. **Don't count child support as debt, deduct it from income (Windows Program: Input in "Family Support Debt").**

Phone/Utility/Checking in Customer Name- Input "Y" if the customer's home phone, cable or utility bill is in their name, or if the customer has a **checking** account **statement** in their name. Bill/Statement **must** come to the customer's address or show service to the customer's address.

Spouse Co-X: Y/N- Input Y only when both spouses sign. Fill in the pop-up questions per policy.

Other Co-X: Y/N- Input Y if there is a non-spouse co-signer. Fill in the pop-up questions per policy.

VEHICLE CLASSIFICATION SHEET 11/2000

IMPORTS

ACURA

Integra Man Trans	3
Legend 86-90	5
Vigor	3
All Others	1

DAEWOO

4 Dr + Auto	3
All Others	4

HONDA

Civic 92-newer 4dr+Auto.	"S"
Civic 92-newer 4dr+Man.	1
Other Civic Automatic	2
Other Civic Man Trans	3
CRX/Prelude	3
Accord 89&Older	3
Accrd 91-94 LX 4dr+Auto	"S"
All Others inc Accrd Wgn.	1

HYUNDAI

Scoupe (All)	5
Other 97+newer	4
All Others	3

ISUZU

Pickups	1
Trooper/Rodeo 4dr+Auto.	2
Trooper/Rodeo Other	3
All Others	5

LEXUS

All	3

MAZDA CARS

MX-6	5
Miata	5
Protégé 94-older	4
RX7	5
929 91-older	4
All Others	3

MAZDA TRUCKS

Pickups Auto+Xcab	2
Navajo	4
All Others	3

MITSUBISHI

Galant 94 & newer	3
Montero	3
Pickups	1
All Others	5

NISSAN CARS

Altima 93-95 w Auto	2
Maxima 89&newer Auto	2
Sentra 92-older	"S"
Sentra 93-newer	1
240SX	4
300 ZX	5
All Others	3

NISSAN TRUCKS

Pathfinder 4 Dr + Auto	1
Pickups	"S"
Quest	2
All Others	3

VEHICLES 10 YRS OLD OR MORE: Add 100,000 miles to odometer if a 5 digit odometer. 6 digit odometer vehicles must be booked with at least 100,000 miles.

IMPORTS (Cont)

TOYOTA CARS

Camry 92-93 Auto	"S"
Celica/Cressida/MR2	3
Corolla 93-94 Auto	"S"
Supra	5
All Others	1

TOYOTA TRUCKS

Pickups	"S"
4-Runner 90-91 V6+4dr+Auto	"S"
Vans 89 & older	4
All Others	1

VOLKSWAGEN

Jetta/Passat 4 Dr.	3
All Others	5

DOMESTICS

BUICK

Quad 4, Tech 4 or	5
Regal 92&newer w 3.8L	2
Other 92&newer w 3.8L	3
Century/Skylark/Regal	3
All Others	4

CHEVROLET

Quad 4, Tech 4 or 2.8L	5
Camaro	5
Corvette	5
Corsica/Caprice	4
All Others	3

CHEVROLET / GMC TRUCKS

Astro/Safari 2WD	1
Blazer 4dr+4.3L 95+	2
S10 Blazer 2dr All	5
C-Series w Auto	1
C-Series Other	2
K-5 Blazer/Tahoe/Yukon	1
Lumina Van	5
S10 X-Cab 4.3L+Auto	1
Suburban	2
All Others	3

CHRYSLER

Cirrus	3
Concorde	4
Town & Country	5
All Others	5

DODGE/PLYMOUTH CARS

Turbos/Convertibles	5
Intrepid	4
Neon 4 dr + Auto	3
Shadow/Sundance	3
Spirit/Acclaim	3
Stratus/Breeze	3
All Others	5

DODGE / PLYMOUTH TRUCKS

Caravan/Voyager 96-newer 2WD	3
Caravan/Voygr Other	5
94+ Trucks V-8	2
Dakota V6/V8	2
All Others	3

DOMESTICS

FORD CARS

Turbo/Supercharger	5
Escort	4
Mustang 94 & newer	2
Taurus Sedan 95 & older.	5
Taurus Wagon	5
T-Bird 90-93	2
All Others	3

FORD TRUCKS

Aerostar 4X4	5
Explorer 4 Dr + Auto	2
Explorer Other	4
F Series Auto + V-8	1
F Series Other	2
Ranger X Cab 6 cyl + Auto	1
Ranger 6 cyl + Auto	2
All Others	3

GEO

Prism 4dr Sedan w Auto	1
Prism 4dr Sedan w Man	2
Tracker	5
All Others	3

JEEP

CJ & Wrangler 6 cyl	1
Other CJ/Wrangler	3
Cherokee 4dr+4.0L+Auto	3
Grand Cherokee	3
All Others	5

LINCOLN

Towncar	4
All Others	5

MERCURY

Capri	5
Tracer	4
Sable Wagon	5
Sable Sedan 95 & older	5
All Others	3

OLDSMOBILE

Quad 4, Tech 4	5
Silhouette	5
All Other 3.8L or V8	4
All Other 4 or 6 cyl	3

PONTIAC

Quad 4, Tech 4	5
Firebird	5
Transport	5
All Others	3

SATURN

All	3

ADDITIONAL POLICIES

1. ANY VEHICLE NOT LISTED SHALL BE CONSIDERED CLASS 5.
2. DO NOT ADD FOR LOW MILES, OR "SOFT ADDS."

WESTLAKE WILL NOT ADVANCE FOR THE FOLLOWING KELLEY ADDS: PREMIUM SOUND, PREMIUM WHEELS, ABS, DUAL AIR BAGS, INTEGRATED PHONE, UPGRADED TOPS, BUMPER, OR PAINT, WIDE/OVERSIZE TIRES, TOW PACKAGE, GRILLE GUARD, WINCH, COMMERCIAL TRUCK ADDS & ANY ITEM NOT IN WORKING ORDER.

3. ANY VARIANCE FOUND BETWEEN ACTUAL & REPRESENTED VALUE OF THE VEHICLE MAY RESULT IN DEALER REPURCHASE.

WESTLAKE FINANCIAL SERVICES

Credit Bureau Input Guidelines

TRW Abbreviation	*(+)*	*(-)*	***no effect***	***TRW Abbreviation***	*(+)*	*(-)*	***no effect***	***TRW Abbreviation***	*(+)*	*(-)*	***no effect***
BK ADJ PLAN		●		CUR WAS 90	●	●		PAID-VOLUSUR	●	●	
BK LIQ REO		●		CUR WAS 90-3+	●	●		PD BY DLER		●	
CHARGE OFF		●		CUR WAS 120-2+	●	●		REDEEMD REPO	●	●	
CLOS NP AA		●		CUR WAS 150-2+	●	●		REFINANC			●
CLOSED-120 2+ TIMES		●		DECEASED			●	REPOSESS		●	
CLOSED-30 2 TIMES		●		DEEDLIEU		●		SCNL		●	
CLOSED-30 3 TIMES		●		FORE PROC		●		SCNL LOC		●	
CLOSED-30 4 TIMES		●		FORECLOS		●		SETTLED	●	●	
CLOSED-30 5 TIMES		●		GOV CLAI M		●		TRANSFER			●
CLOSED-30 6+ TIMES		●		INACTIVE			●	TRMDFALT		●	
CLOSED-30 DAY DEL		●		INQUIRY			●	VOLUSURR		●	
CLOSED-30 WAS 60		●		INS CLAIM			●				
CLOSED-60 2 TIMES		●		NO STATUS			●	BK 11-DISCHG			●
CLOSED-60 3 TIMES		●		NOT PAY AA		●		BK 11-DISMIS			●
CLOSED-60 4+ TIMES		●		OPEN-30 2 TIMES		●		BK 11-PETIT			●
CLOSED-90 2 TIMES		●		OPEN-30 3 TIMES		●		BK 12-DISCHG			●
CLOSED-90 3+ TIMES		●		OPEN-30 4 TIMES		●		BK 12-PETIT			●
CLOSED-90 WAS 120+		●		OPEN-30 5 TIMES		●		BK 13-DISCHG			●
CLOSED-DEL WAS 120+		●		OPEN-30 6+ TIMES		●		BK 13-DISMIS			●
CLOSED-DEL WAS 90		●		OPEN-30 DAY DEL		●		BK 13-PETIT			●
CLOSED-DELIQ 120		●		OPEN-30 WAS 60		●		BK 7-DISCHG			●
CLOSED-DELIQ 150		●		OPEN-90 WAS 120+		●		BK 7-DISMIS			●
CLOSED-DELIQ 180		●		OPEN-DEL WAS 120+		●		BK 7-PETIT			●
CLOSED-DELIQ 60		●		OPEN-DEL WAS 90		●		CH SUP JUDG		●	
CLOSED-DELIQ 90		●		OPEN-DELINQ 120		●		CH SUP SATIS			●
COLLACCT		●		OPEN-DELINQ 150		●		CITY LIEN		●	
CR CD LOST			●	OPEN-DELINQ 180		●		CITY LIEN REL			●
CR LN CLOSED			●	OPEN-DELINQ 60		●		CIV CL JUDG		●	
CR LN REINST	●			OPEN-DELINQ 90		●		CIV CL SATIS			●
CUR ACCT	●			PAID	●			CIV CL VACAT			●
CUR WAS COLL	●	●		PAID-30 DAY DEL	●			CO LIEN REL			●
CUR WAS FORE	●	●		PAID-90 WAS 120+	●	●		COUNTY LIEN		●	
CUR WAS 120	●	●		PAID-CHARGOFF	●	●		FED TAX LIEN		●	
CUR WAS 150	●	●		PAID-COLLACCT			●	FED TAX REL			●
CUR WAS 180	●	●		PAID-CURR ACCT	●			MECH LIEN		●	
CUR WAS 30	●			PAID-DEL WAS 120+	●	●		MECH LN REL			●
CUR WAS 30-2	●			PAID-DEL WAS 90	●	●		SM CL JUDGMT		●	
UR WAS 30-3	●			PAID-DELIQ 120	●	●		SM CL SATIS			●
UR WAS 30-4	●			PAID-DELIQ 150	●	●		SM CL VACAT			●
UR WAS 30-5	●			PAID-DELIQ 180	●	●		STATE TX LN		●	
UR WAS 30-6+	●			PAID-DELIQ 60	●	●		STATE TX REL			●
UR WAS 60	●	●		PAID-DELIQ 90	●	●		SUIT DISMISS			●
UR WAS 60-2	●	●		COFF NOW PAY		●		SUIT FILED		●	
UR WAS 60-3	●	●		PAID-FORECLOS	●	●		WAGE ASSIGN		●	
UR WAS 60-4+	●	●		PAID-REPOSES	●	●		W/A RELEASED			●

WEST .AKE FINANCIAL S_RVICES

Credit Bureau Input Guidelines

NOTE: ANY ACCOUNT WITH A PAST DUE AMOUNT IS A DEROG

Account Status	*Last Entry in Account History*	*Highest Deliq # in Account History*	*Meaning*	*(+)*	*(-)*	*No Effect*
(No #)	No History	No History	No Status			●
0	No History	No History	New Acct			●
1	No History	No History	Curr or Paid AA	●		
1	*	2	Curr/Paid Was 30	●		
1	*	3	Curr/Paid Was 60	●	●	
1	*	4	Curr/Paid Was 90	●	●	
1	*	5	Curr/Paid Was 120	●	●	
2	*	Any	Curr/Paid Was 30	●		
2	2	Any	30 Day Delinquent		●	
3	*	Any	Curr/Paid Was 60	●	●	
3	2 or 3	Any	Now Delinquent Was 60		●	
4	*	Any	Curr/Paid Was 90	●	●	
4	2 or 3 or 4	Any	Now Delinquent Was 90		●	
5	*	Any	Curr/Paid Was 120	●	●	
5	2,3,4,5	Any	Now Delinquent Was 120		●	
7	Any	Any	Paying under Plan			●
8	Any	Any	Repo		●	
9	None	None	Charge Off		●	
9	CHARGED OFF		Charge Off		●	
9	PAID CHARGE OFF		Paid Charge Off	●	●	
	SOLD		Sold to Other Lender			●
	TRANSFERRED		Account Transferred			●
	REFINANCED		Account Refinanced			●
PUBLIC RECORDS/OTHER INFORMATION ABBREVIATIONS						
JUDG			Judgment		●	
ST JUDG			Satisfied Judgment			●
BKRPT			Bankruptcy			●
WEP			Wage Earner Plan			●
FORCL			Foreclosure		●	
DV FD			Divorce Filed			●
DV FL			Divorce Final			●
SP MT			Separate Maintenance			●
N/RES			Non-Responsibility			●
GARN			Garnishment			●
LIEN			Tax Lien		●	
LIEN		+ RELEASED	Tax Lien Released			●
NPFC			Non-Prof Fin Counsel			●
FN ST			Financial Statement			●
SECLN			Secured Loan			●
COLL			Collection Account		●	
COLL		+ PAID	Paid Collection Acct			●

HOW TO SCORE CREDIT (IN A NUTSHELL--ASK REP FOR SPECIFIC QUESTIONS)

1. Paid or Current credit never more than 30 days late is 1 good.
2. Paid or Current credit that was 60 or more days late is 1 good and 1 derog.
3. Any delinquent account or unpaid charged off account is 1 derog.
4. Collection accounts are derog; paid Collection accounts do not count as anything.
5. Tax Liens and Judgments are derog; if satisfied they do not count.
6. Student Loans and Child Support do not count, BUT we do count the debt from child support.
7. Transferred, Sold, or Refinanced lines of credit do not count.
8. Multiple lines of credit from the same creditor normally count as 1 line of credit.

CBI 6-99

Westlake
FINANCIAL SERVICES

PREFERRED PROGRAM GUIDELINES

Effective Date: 11-15-2000

INSTRUCTIONS:

1. Score credit as normal with the standard program guidelines with the following exceptions:
   a. **Paid Charge Off. Count as NO EFFECT.**
   b. **Settled. Count as NO EFFECT.**
   c. **Medical Collections. Count as Unpaid Collection Accounts (derog).**
2. We will not ask you for # Derog Credit Items. Instead we will ask you about **UNPAID** Charge Offs and Collection Accounts (see guidelines below). Therefore, any account that we normally consider to be a +/- is a + only in this program. Exceptions to this are **Paid Charge Off and Settled.** These accounts will be considered as NO EFFECT. Also for this program, we WILL count Medical Collections as Unpaid Collection Accounts.
3. Input Yrs on Credit Bureau, # Good, and $ Hi Good as normal. The questions # Unpaid Coll Accts, # Unpaid Chg Offs, and $ Hi Unpaid Chg/Coll are new. Score those using the following guidelines:

# Unpaid Coll Accts / # Unpaid Charge Offs: Do not count unpaid accounts charged off during bankruptcy. If the BK was not completed, then you count them. What we are looking for are the bad accounts for which the customer is **still liable.**

$ Hi Unpaid Chg Off/Coll: This is the high $ derog on an account that is a charge off or collection. What we are looking for is the highest dollar amount of a Collection Account or Charge Off for which the customer is **still liable.** So you would not count BK accounts either for this question.

OTHER POLICIES:

- **No Prior Repos/Auto Losses, INCLUDING with/before BK**
- **No Multiple Bankruptcies or Multiple BK Filings**
- **No open (active) delinquencies**
- **No TMU, Salvage, Exempt, or Other Branded Titles**
- **Max of 1 unreported Paid Auto and 1 unreported other loan**
- **NO Rent-To-Own accounts or Pre-Paid phone**
- **Vehicles Class 1-4 only**
- **Service Contracts must cover at least 1/2 the term of the contract**
- COUNT MEDICAL COLLECTIONS AS UNPAID COLLECTION ACCOUNTS!
- **PAID CHARGEOFFS AND SETTLED ACCOUNTS DO NOT COUNT AS CREDIT!**
  (They still count for Time on Bureau)

STIPS:

- **Complete** application; incomplete apps will be **returned**
- Minimum 4 different references, POI, copy of DL **from state of residence** (MUST have)
- Must have copy of phone/util bill if input "Y" in program for Ph/Util Bill
- NO 30 day insurance binders or polices or dealer-procured/sponsored insurance
- UIC 6 month policy OK

HINTS!!!

- **When the program says "SORRY," it will give you a reason. Follow the directions.**
- You can and should manipulate down, price, and term to make your deal.
- Must have at least 2 years Credit History (or a solid Cosignor), 4 years helps a lot.
- The program will accept very high debt ratios if the debt is from Rent or Mortgage.
- Equity is key! Don't expect to max out the amount financed every time. If the program won't give you the approval when you max it out, you are going to have to adjust the down/price to see how far it is willing to go.

WESTLAKE FINANCIAL SERVICES
PROGRAM INSTRUCTIONS

__Credit Application__ - Must contain landlord name and phone number, 5 yrs job and residence history, bank account info, and relative reference (Mother, Father, Sister, Brother). Failure to provide the above will result in a TD. Applications found to be falsified will be a TD. It is the dealership's responsibility to verify the information on the credit application prior to submission for funding.

__Contract__ – Contracts are to be written at the rate indicated by the Buy Program. **CA ONLY: Simple Interest contracts to be written at 23.9% APR.** Must sign in designated "assignment" area on front and back of contract, or it will be returned immediately.

__Phone Bill__ - Bill must go to customer's residence. Westlake cannot purchase a deal if the customer's phone bill goes to another address. Deals **will not** be purchased without documentation showing the **address** the number goes to. **ALL PAGES OF PHONE BILL ARE REQUIRED.** ***In areas that the local phone company doesn't put the address on the phone bill SOLID PROOF of residence for the person named on the phone bill besides a phone bill is required.*** If the phone is prepaid or from a non-major company, then must be able to prove that the phone has been in service at least 3 months. If no land-line phone, then a complete cell phone bill in customer's name is ok if it goes to physical address and customer has a utility bill or checking account statement in name going to customer's physical address. **Pre-paid cell is unacceptable.** Phone bills past due more than the Westlake payment or that are actually disconnect notices are the same as not having a phone.

__Open Auto Loans__ – 1 pre-existing open auto loan allowed if married and both sign. Cox may have an open auto. No open auto if single buyer.

__Pick Payments__ - Up to 25% of the total down, up to $500. Last pick due no later than 14 days prior to 1st payment.

__Military__ - Rank E3+ ONLY. Term of Contract cannot exceed six (6) months beyond separation date listed on LES. Deal must arrive with **completed** MAC allotment and completed burnout or it will be returned immediately.

__Credit Counseling (CCC)__ - Westlake will not purchase contracts with buyers who are in Credit Counseling (CCC) or have accounts presently being managed by a Credit Counseling service.

__Present or Prior Westlake Accounts__ - Westlake will allow a second account on a couple who have made at least 6 payments on an account that is paid up-to-date. Any Westlake deficiency must be paid in full before considering a new Westlake deal.

__Non-Reporting Good Accounts__ - Max of one auto loan and one other account. **NO** rental, medical, or dental.

__Derog on Bureau-Now Paid__ - Chargeoffs and coll accts must have been paid at least 30 days prior to dealer running bureau. Delinquent accounts that are now current must have been paid up-to-date prior to dealer running bureau.

__Medical Collection Accounts__ - Do not count Collection Accounts which are medical in nature. In order to be considered a Medical Collection, the account must reflect **on the bureau** a Doctor, Hospital, Radiologist, Emergency Room, X-Ray Facility/Tech, or other entity that is **clearly** Medical or Dental. The definition of "clearly" is left solely to the discretion of the Westlake buyer, so use common sense in applying this rule.

WESTLAKE FINANCIAL SERVICES

PROGRAM INSTRUCTIONS

Utility Bills - A utility is tap water, gas, electric, cable, satellite connection, garbage, or sewer. It is **NOT** jugs of water or newspaper.

Dealer Employees - Any deal on **any** employee of any dealer must be pre-approved by Westlake prior to submission.

TMU/Salvage Vehicles - TMU is acceptable with a Statement of Facts from the customer acknowledging TMU. For the book and program input, add 100,000 miles to the odometer. **NO SALVAGE TITLES, LEMON-LAW TITLES, POLICE/TAXI/FIRE/FLOOD TITLES, OR ANY OTHER BRANDED TITLES. ODOMETER FORMS MUST ACCOMPANY DEAL. NO PURCHASES WILL BE MADE WITH MILEAGE REPRESENTED AS "EXEMPT".**

Older Units - Vehicles 10 years old or more are assumed to have over 100,000 miles on them, and should be booked out that way. A 1988 car with a 5 digit odometer that says 48,000 miles must booked out as 148,000 miles. A 1988 car with a 6 digit odometer showing less than 100,000 miles must be booked out as 100,000 miles. No exceptions.

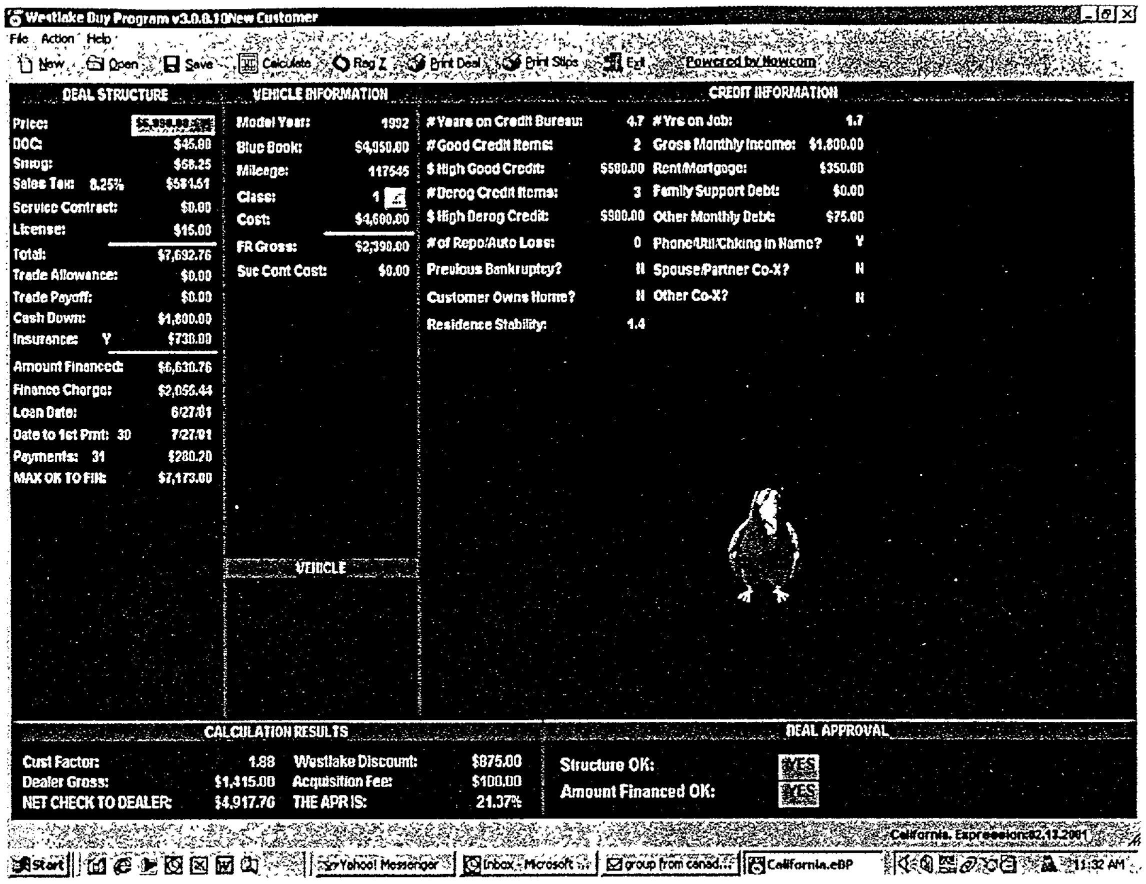
How we count Time Residence – Life begins at 18. If the credit bureau shows a different address with updates, then allow no more than 1 month after the earlier update. Use the latest different address for updates. If the driver's license has a different address, allow no more than 1 month after the date of the license. Use the lower of this number or what the application says. These policies hold even if the application shows a longer time at residence, or the credit bureau shows the current address before the different address. No documentation will **ever** change this policy.

Booksheet – **KELLEY WHOLESALE: DO NOT ADD FOR LOW MILEAGE WHEN BOOKING THE VEHICLE.** Do not add for the following options: Premium Sound, Premium Wheels, ABS, Dual Air Bags, Integrated Phone, Imitation/Padded/Vinyl tops, Custom Bumper, 2 tone paint, Wide/Oversize tires, tow package, Winch, Snow Plow, commercial truck adds and any item not in working order. Any variance discovered between actual & represented valuation of the vehicle by the dealership may result in dealer repurchase.

Rent - If the customer lives with relatives or pays no rent, use the greater of 10% of Gross Income or **$250** for rent.

Open Delinquencies - Westlake will not purchase a deal if the buyer has more than 2 open delinquencies on Credit Bureau. An open (active) delinquency is defined as any account counted as derog only that is not a chargeoff, coll acct, or BK liquidation. An account that is closed but is also currently delinquent and not a chargeoff is an open delinquency. If *Home Loan* is currently delinquent *DEAL WILL BE AN AUTOMATIC DECLINE*.

Our Philosophy - Westlake believes in a "Win-Win" approach for both the dealer and Westlake. We believe that our program allows the dealer more flexibility in structuring a deal than any other Finance Company program. We put a great deal of trust in our dealers when we give them access to our buy model, and we expect them to make deals that make sense. While we have sometimes accepted soft documentation on strong deals, we expect strong documentation on weak deals. Therefore, while we will make every effort to fund all deals that are approved by the Buy Program, it is ultimately up to you, our dealer, to ensure that the deals you send in have integrity and are fair for all parties concerned.



Westlake Buy Program v3.0.0.10New Customer

File Action Help

New Open Save Calculate Reg Z Print Deal Print Slips Exit Powered by Nowcom

DEAL STRUCTURE

Price:		$5,990.00
DOC:		$45.80
Smog:		$68.25
Sales Tax:	8.25%	$581.51
Service Contract:		$0.00
License:		$15.00
Total:		$7,692.76
Trade Allowance:		$0.00
Trade Payoff:		$0.00
Cash Down:		$1,800.00
Insurance:	Y	$730.00
Amount Financed:		$6,630.76
Finance Charge:		$2,056.44
Loan Date:		6/27/01
Date to 1st Pmt:	30	7/27/01
Payments:	31	$280.20
MAX OK TO FIN:		$7,173.00

VEHICLE INFORMATION

Model Year:	1992
Blue Book:	$4,050.00
Mileage:	117545
Class:	1
Cost:	$4,600.00
FR Gross:	$2,390.00
Svc Cont Cost:	$0.00

VEHICLE

CREDIT INFORMATION

# Years on Credit Bureau:	4.7	# Yrs on Job:	1.7
# Good Credit Items:	2	Gross Monthly Income:	$1,800.00
$ High Good Credit:	$580.00	Rent/Mortgage:	$350.00
# Derog Credit Items:	3	Family Support Debt:	$0.00
$ High Derog Credit:	$900.00	Other Monthly Debt:	$75.00
# of Repo/Auto Loss:	0	Phone/Util/Chking in Name?	Y
Previous Bankruptcy?	N	Spouse/Partner Co-X?	N
Customer Owns Home?	N	Other Co-X?	N
Residence Stability:	1.4		

CALCULATION RESULTS

Cust Factor:	1.88	Westlake Discount:	$875.00
Dealer Gross:	$1,415.00	Acquisition Fee:	$100.00
NET CHECK TO DEALER:	$4,917.76	THE APR IS:	21.37%

DEAL APPROVAL

Structure OK:	YES
Amount Financed OK:	YES

California. Expression:02.13.2001

Start | Yahoo! Messenger | Inbox - Microsoft ... | group from canad... | California.eBP | 11:32 AM

APPENDIX - B

COMPUTER PROGRAM

```
'<%Template=California%> <%Version=Jay Test%>
' Template=California
' California Expression Template
' Modification Date : Nov 16, 2000
' Reason: converted from Delphi to VB Script
' Modification Date : Nov 17, 2000
' Reason: Added code for COM, modified for Stand Alone BP
' Modification Date : Nov 22, 2000
' Reason: Added TotalofPayments calculation
' Modification Date : Jan 25, 2001
' Reason: Added insuarnce cap beyond $10,000.00
' Modification Date : Feb 13, 2001
' Reason: Repaired wizard re o/a, etc
' Modification Date : Feb 26, 2001 - John Sun
' Reason: Added error handling - when error occurs, system need to
continue and trap
' all the error messages.
' Modification Date : Mar 26, 2001 - Mike Duke
' Reason: Repaired Ins Lookup Table to account for all Carryback
possibilities.
' Modification Date : Apr 03, 2001
' Reason: Made minimum Total Income = $1.00
' Modification Date : Apr 09, 2001
' Reason: Move Big Mile Hit expressions in proper order for proper
recalc when opening saved deal
' Modification Date : Apr 30, 2001
' Reason: Fix error in Job Lookup Table
' Modification Date : May 16, 2001
' Reason: Fix error in CF Scaler Lookup
' Modification Date : May 23, 2001
' Reason: Allow Class 5 for reserve deals
'------------------------------Added for Stand Alone
BP-------------------------------


'On Error Resume Next

Set BPMod = CreateObject("BPfunctionsModule.BPFunctions")
'-------------------------------------------------------------------
--------------------

'[CONSTANTS]

'System Error
DIM SystemError
SystemError = ""

Acqfee=100
'Note DealerGross, NetCheckToDealer

TooSmallPmt = 140
'Note DebtAdjustment Model, Final Reserve, Error Section

MinDiscount = 0.10
'Note MinDerog, MinBK, MinFact, FinalReserve

CurrYear = 2000
'Note MaxCB Model, CarAge variable


'*******************************************************************
***
'CarYear = vYear
'*******************************************************************
***
if (vYear < 5) then
  CarYear = vYear + 2000
  else if (vYear < 100) then
    CarYear = vYear + 1900
    else
      CarYear = vYear
  end if
end if
'Note
'       CarYear is used in MaxCB, CarAge var, Error Section
'*******************************************************************
***


'CHANGE
hint =""
hint1=""
hint2=""
hint3=""
hint4=""
hint5=""
hint6=""
hint7=""
hint8=""
hint9=""
hint10=""
hint11=""
hint12=""
hint13=""
hint14=""
hint15=""
hint16=""
hint17=""
hint18=""
hint19=""
hint20=""
hint21=""
hint22=""
hint23=""
hint24=""
hint25=""




'*******************************************************************
***
'Deal Structure Calculation Area
'Calculate TaxAmount And SubTotal
'*******************************************************************
***
'LOOK AT THIS TAXRATE AS COMPARED TO DUKES FILE
'Input Parameters for this block are basically input variables
' TaxRate, Price, Smog, Doc, SmogCert, Tax, LicFee, Warr, Down,
TradeAllowance
' TradePayoff

Tax = (TaxRate/100) * (Price + Smog + Doc)
SubTot = cdbl(Price + Doc + Smog + SmogCert + Tax + LicFee + Warr)
TotalDown = Down + TradeAllowance - TradePayoff
TotalLessIns = SubTot - TotalDown

'Note these are all variables not models
'Tax used in SubTot
'SubTot used in TotalLessIns variable, CB variable
'TotalDown used in TotalLessIns variable, CB variable, BKBonus,
'       FTBBonus & SmallFTBBonus, and MINBK
'TotalLessIns used in Ins variable, EquityTest variable, HICBHIT,
'       OptimalCB Credit, FineTune, FinalCustomerFactor,
CFPHBillScaler in the
'       Final CF calculation, DebtScaler, SpreadNum, MinFact,
FinalReserve, Error section
'*******************************************************************
***
```

10045975.010602

```
'********************************************************************
***
'Calculate Insurance Amount If needed
'********************************************************************
***
'Input Parameters for this block are basically input variables
' InsFlag, TotalLessIns

if (InsFlag = 1) then
  if (TotalLessIns <= 10000)  then
    Ins = LookupIns(TotalLessIns, 2 )
  else
    Ins = 0.1088*TotalLessIns+95
  end if
else
  Ins = 0.00
end if

'Note
'INS is used in many places
'       CB variable, MaxCB, EquityTest variable, TotalDebt, FTBBonus
& SmallFTBBonus,
'       HICBHIT, ExcessTerm Debt, Xterm, SpreadNum, CheckToDealer,
Error Section
'********************************************************************
***



'********************************************************************
***
' This is the amount Financed
'********************************************************************
***
'Input Parameters for this block are basically input variables
' SubTot, TotalDown, Ins

CB = (SubTot - TotalDown) + Ins

'Note this variable is used in many places
'       Payment, IntCost Var, Add-On Var, SigDown, DebtAdjustment,
TotalDebt,
'       FTBBonus, SmallFTBBonus, HICBHit, OptimalCBHit, ExcessTerm,
SpreadNum,
'       RealOA var, CheckToDealer var, AmtOk var, Error Section
'********************************************************************
***



'********************************************************************
***
'Calculate APR --- Lookup Interest Rate
'********************************************************************
***
'Input Parameters for this block are basically input variables
' Term

Interest = LookupApr( Term )
APR = Interest

'Note Interest is used in many places
'       APR, DebtAdjustment, TotalDebt
'********************************************************************
***
```

```
'**********************************************************************
***
'Calculate Payment
'**********************************************************************
***
'Input Parameters for this block are basically input variables
' CB, Term, DaysToPay

PaymentA = BPMod.bp_AddOnPMT( CB, Term, 0.12, DaysToPay )
Payment = BPMod.bp_Trunc( PaymentA,2 )

'Note payment used in a lot of places
'        IntCost var, Add-On var, TotalOfPayments, DebtAdjustment,
TotalDebt,
'        FTBBonus, OptimalCBCredit, Excess Term, Payment Probability,
PPAdjust,
'        SpreadNum, FinalReserve
'**********************************************************************
***


'**********************************************************************
***
'ADDON is the total dollar amount of Interest
'**********************************************************************
***
'Input Parameters for this block are basically input variables
' Payment, Term, CB, Price, Cost, Reserve, Warr, WarCost, AcqFee
'These are all output variables calculated

IntCost = ( Payment * Term ) - CB
AddOn = Payment * Term - CB
TotalofPayments= Payment*Term
FrGross = Price - Cost
DealerGross=PRICE-COST-RESERVE+WARR-WARCOST-AcqFee

'Note
'        IntCost, AddOn, TotalOfPayments, FrGross are used basically
outputs
'        DealerGross is used in Error Section
'**********************************************************************
***


'**********************************************************************
***
'        Max Amount Financed Calculation Area = "MaxCB"
'**********************************************************************
***
'**********************************************************************
***
'Calculate Hit For very high miles = BigMileHit used to calculate
MAXCB
'**********************************************************************
***
'Input Parameters for this block are basically input variables
' vClass, Miles

BigMilesStart = 185000
BigMilesRange_1 = 50000
BigMilesRange_2 = 50000
BigMilesRange_3 = 50000
HitBigMiles_1 = 0.15
HitBigMiles_2 = 0.15
HitBigMiles_3 = 0.15

LotsOfMiles_1 = BigMilesStart - ( vClass * 10000 )
LotsOfMiles_2 = LotsOfMiles_1 + BigMilesRange_1
```

```
LotsofMiles_3 = LotsOfMiles_2 + BigMilesRange_2

HitRate_1 = ( HitBigMiles_1 + ( vClass / 100 ) ) / BigMilesRange_1
HitRate_2 = ( HitBigMiles_2 + ( vClass / 100 ) ) / BigMilesRange_2
HitRate_3 = ( HitBigMiles_3 + ( vClass / 100 ) ) / BigMilesRange_3

BigMileDelta_2 = BPMod.bp_MIN( Miles - LotsOfMiles_2, BigMilesRange_2
) * HitRate_2
BigMileDelta_3 = BPMod.bp_MIN( Miles - LotsOfMiles_3, BigMilesRange_3
) * HitRate_3
BigMileHit_1 = BPMod.bp_MIN( Miles - LotsOfMiles_1, BigMilesRange_1 )
* HitRate_1
BigMileHit_2 = BPMod.bp_IFG( Miles, LotsOfMiles_2, BigMileHit_1 +
BigMileDelta_2, BigMileHit_1 )
BigMileHit_3 = BPMod.bp_IFG( Miles, LotsOfMiles_3, BigMileHit_2 +
BigMileDelta_3, BigMileHit_2 )
BigMileHit = BPMod.bp_IFG( Miles, LotsOfMiles_1, BigMileHit_3, 0 )

'Note
'       BigMileHit is the output used in calculating MaxCB
'*********************************************************************
***


'*********************************************************************
***
'Calculate Regular Hi Mile Hit = "HiMileHit"
'*********************************************************************
***
'Input Parameters for this block are basically input variables
' vClass, Miles, Book

MCBHiMiles = 140000
MCBHiMilesRange = 10000

OverMiles = MCBHiMiles - MCBHiMilesRange
MaxHiMileHit = LookupTermTable( vClass, 10 )
MCBHitRate = MaxHiMileHit / MCBHiMilesRange
HiMileHitExp1 = BPMod.bp_MIN( ( Miles - OverMiles ), MCBHiMilesRange
) * MCBHitRate * Book
HiMileHit = BPMod.bp_IFG( Miles,MCBHiMiles -
MCBHiMilesRange,HiMileHitExp1, 0 )

'Note
'       HiMileHit is the output variable used in calculating MaxCB
'*********************************************************************
***

'*********************************************************************
***
'Calculate WarrAllowance
'Note I thought that this variable could stand by itself
'*********************************************************************
***
'Input Parameters for this block are basically input variables
' Warr

MaxWarrCB = 250

WarrAllowance = BPMod.bp_MIN( MaxWarrCB, Warr )

'Note WarrAllowance is used in MaxCB, SigDown, TotalDebt, Excess
Term,
'       SpreadNum, MinFact, FinalReserve, Error Section
'*********************************************************************
***


'*********************************************************************
***
'Calculate MaxCB
```

```
'*********************************************************************
***
'Input Parameters for this block are basically input variables
' vClass, Book, HiMileHit, WarrAllowance, Ins, BigMileHit, CurrYear,
CarYear

BMHiLimit = 6000
BMLowLimit = 2000
MCBMaxIns = 1000

BMRange = BMHiLimit - BMLowLimit
CarClassAdv = LookupTermTable( vClass, 8 ) * Book
MaxBookAdv = LookupTermTable( vClass, 9 ) + Book
PossibleAdv = CarClassAdv - HiMileHit + WarrAllowance + BPMod.bp_MIN
( Ins, MCBMaxIns )
OKAdv = BPMod.bp_MIN( PossibleAdv, ( MaxBookAdv + Ins + WarrAllowance
) )
BigMileSmackScaler = BPMod.bp_MAX( BPMod.bp_MIN( ( OKAdv - Ins -
BMLowLimit ) / BMRange, 1 ),0 )
BigMileSmack = ( OKAdv - Ins ) * BigMileHit * BigMileSmackScaler
MaxAltCB = 1500 + Ins - 100 * ( CurrYear - BPMod.bp_MIN( CurrYear,
CarYear ) - 10 )
MaxCB = BPMod.bp_MAX( ( OKAdv - BigMileSmack ), MaxAltCB )

'Note
'       MaxCB is used in EquityTest var, SigDown, FineTune, Xterm,
RealOA,
'       Error Section
'*********************************************************************
***
'*********************************************************************
***
'                                End MaxCB
'*********************************************************************
***


'*********************************************************************
***
'Assorted One Line Variables for future use
'*********************************************************************
***
'Input Parameters for this block are basically input variables
' Down, TradeAllowance, TradePayoff, TradeScaler, CurrYear, Caryear,
' TotalLessIns, MaxCB, Ins
' These again are some variable which are later used.

TradeScaler=0.70

RealDown = Down + ( TradeAllowance - TradePayoff ) * TradeScaler
CarAge = CurrYear - CarYear
EquityTest = TotalLessIns / ( MaxCB - Ins )

'Note
'       RealDown used in SigDown, OptimalCBCredit, Error Section
'*********************************************************************
***


'*********************************************************************
***
'Calculate Good/Derog including Spouse
'*********************************************************************
***
'Input Parameters for this block are basically input variables
' Spouse, Good, SpGood, Derog, SpDerog, HiGood, SpHiGood, HiDerog,
SpHiDerog

TotalGood = BPMod.bp_IFB( Spouse,( Good + SpGood ) / 2, Good )
TotalDerog = BPMod.bp_IFB( Spouse, ( Derog + SpDerog ) / 2, Derog )
RealHiGood = BPMod.bp_IFB( Spouse, BPMod.bp_MAX( HiGood, SpHiGood ),
HiGood )
RealHiDerog = BPMod.bp_IFB( Spouse, BPMod.bp_MAX( HiDerog, SpHiDerog
), HiDerog )

'Note
'       TotalGood used in RealINC, GoodScaler/BadScaler, BKBonus,
HICBHIT,
'               FineTune, FinalCFCalculation, DEBTScaler, MinBK
'       TotalDerog used in RealINC, GoodScaler/BadScaler, HiCBHit,
FineTune,
'               FinalCFCalculation, DebtScaler
'       RealHiGood used in GoodScaler/BadScaler, BKBonus, HiCBHit,
'               FinalCFCalculation, MinBK
'       RealHiDerog used in GoodScaler/BadScaler,
FTBBonus/SmallFTBBonus,
'               HiCBHit, FineTune, FinalCF, DebtScaler, MinDerog.
'*********************************************************************
***

'*********************************************************************
***
'Calculate TotalINC
'Note I separated because I thought it could be calculate by itself
and
'does have to be included in RealInc as it used in many places
'*********************************************************************
***
'Input Parameters for this block are basically input variables
' Spouse, Inc, SpInc, Support

TotalInc=BPMod.bp_MAX(BPMod.bp_IFB(Spouse, Inc+SpInc-Support,
Inc-Support), 1)

'Note
'       TotalInc in RealInc, RealJob
'*********************************************************************
***


'*********************************************************************
***
'Calculate Income including Spouse = "RealINC"
'*********************************************************************
***
'Input Parameters for this block are basically input variables
' TotalGood, TotalDerog, YrsTRW, Repos, Inc, Support, SpInc,
TotalInc, Spouse

'TotalInc = BPMod.bp_IFB( Spouse, (Inc + SpInc - Support), (Inc -
Support))
'TotalInc=BPMod.bp_MAX(BPMod.bp_IFB(Spouse, Inc+SpInc-Support,
Inc-Support), 1)
RealIncCond1 = BPMod.bp_IFG( TotalGood, 1.5, 1, 0 )
RealIncCond2 = BPMod.bp_IFL( TotalDerog, TotalGood, 1,0 )
RealIncCond3 = BPMod.bp_IFGE( YrsTRW, 2, 1,0 )
RealIncCond4 = BPMod.bp_IFLE( TotalDerog, 2, 1, 0 )
RealIncCond5 = BPMod.bp_IFE( Repos, 0, 1,0 )
RealIncCond = RealIncCond1 * RealIncCond2 * RealIncCond3 *
RealIncCond4 * RealIncCond5
MinInc = BPMod.bp_MAX( Inc-Support, SpInc-Support )
IncHit = BPMod.bp_MAX( 1 - ( TotalInc / 10000 ), 0.75 )
RealIncExp2 = BPMod.bp_MAX( BPMod.bp_MAX( TotalInc * IncHit, TotalInc
- 500 ), MinInc )
RealIncExp1 = BPMod.bp_IFB( RealIncCond, TotalInc, RealIncExp2 )
RealIncExp = BPMod.bp_IFB( Spouse, RealIncExp1, TotalInc )
RealInc = BPMod.bp_Max( RealIncExp, 1 )
'Two lines above are there in the new dukes file
'RealInc = BPMod.bp_IFB( Spouse, RealIncExp1, TotalInc )

'Note
'       RealInc is used in CountRent/CrapRatio,
FTBBonus/SmallFTBBonus,
'            TotalDebt, FinalCFCalculation, DebtScaler,
DebtProblem, MinBK, Error Section
'*********************************************************************
***


'*********************************************************************
***
'CoxScaler is to be used if Cox=Yes
'*********************************************************************
***
'Input Parameters for this block are basically input variables
' CoxGood, Good, CoxDerog, CoxRepo, CoxInc, CoxHome, CoxParent,
YrsTRW
' Derog

GoodCreditExp = BPMod.bp_IFL( CoxGood,Good, - 2,0 )
GoodCreditPoints = BPMod.bp_IFB( BPMod.bp_IFG( CoxGood,Good,1,0 ) *
BPMod.bp_IFGE( CoxGood,4,1,0 ), 2, GoodCreditExp )
DerogExp = BPMod.bp_IFB( BPMod.bp_IFG( CoxDerog,3,1,0 ) +
BPMod.bp_IFG( CoxDerog,CoxGood,1,0 ), - 1, 0 )
DerogCreditPoints = BPMod.bp_IFB( BPMod.bp_IFLE( CoxDerog,CoxGood *
0.5, 1,0 ) * BPMod.bp_IFLE( CoxDerog,3,1,0 ) * BPMod.bp_IFGE(
CoxGood,1,1,0 ), 2, DerogExp )
RepoPoints = BPMod.bp_IFE( CoxRepo,0,1, - 10 * CoxRepo )
IncAccounts = BPMod.bp_MAX( (CoxGood + CoxDerog), 1 )

IncDivAcct = BPMod.bp_IFE( ( CoxGood + CoxDerog ), 0, 0, (CoxInc /
IncAccounts) )
IncomePointsElseExp = BPMod.bp_IFB( BPMod.bp_IFGE( IncDivAcct,200,1,0
) + BPMod.bp_IFGE( CoxInc,4000,1,0 ), 3, ( ( IncDivAcct - 100 ) / 100
) * 3 )
IncomePoints = BPMod.bp_IFLE( IncDivAcct,100,0, IncomePointsElseExp )

CoxOwnHomePoints = BPMod.bp_IFB( CoxHome,3,0 )
CoxParentOfBuyerPoint = BPMod.bp_IFB( CoxParent,5, - 1 )
BuyerLowOnBureauPointElseExp2 = BPMod.bp_IFLE( YrsTRW,3,0, - 1 )
BuyerLowOnBureauPointElseExp = BPMod.bp_IFLE( YrsTRW,2,1,
BuyerLowOnBureauPointElseExp2 )
BuyerLowOnBureauPoint = BPMod.bp_IFLE(
YrsTRW,1,3,BuyerLowOnBureauPointElseExp )
CoxPoints = GoodCreditPoints + DerogCreditPoints + RepoPoints +
IncomePoints + CoxOwnHomePoints + CoxParentOfBuyerPoint +
BuyerLowOnBureauPoint

GoodCoxExp1 = BPMod.bp_IFOR2( BPMod.bp_IFAND2( BPMod.bp_IFGE(
CoxInc,1500,1,0 ), BPMod.bp_IFGE( IncDivAcct,300,1,0 ), 1, 0 ),
BPMod.bp_IFGE( CoxInc,2000,1,0 ), 1, 0 )
GoodCoxExp2 = BPMod.bp_IFE( CoxRepo, 0, 1,0 )
GoodCoxExp3 = BPMod.bp_IFL( CoxDerog, 3, 1, 0 )
GoodCoxExp4 = BPMod.bp_IFOR2( BPMod.bp_IFAND2( BPMod.bp_IFGE(
CoxGood,5,1,0 ), BPMod.bp_IFGE( CoxGood,5 * CoxDerog, 1,0 ),1,0
),BPMod.bp_IFAND2( BPMod.bp_IFB( CoxHome,1,0 ), BPMod.bp_IFLE(
CoxDerog, 1, 1, 0 ),1,0 ), 1,0 )
'GoodCoxInc = BPMod.bp_IFB( GoodCoxCond, BPMod.bp_MIN( ( CoxInc -
1500 ) / 1000, 1 ),0 )
'Mike Duke's new program has it correct.
GoodCoxCond = GoodCoxExp1 * GoodCoxExp2 * GoodCoxExp3 * GoodCoxExp4
GoodCoxInc = BPMod.bp_IFB( GoodCoxCond, BPMod.bp_MIN( ( CoxInc - 1500
) / 1000, 1 ),0 )
DerogNotZero = BPMod.bp_MAX( CoxDerog, 1 )
GoodCoxCredit = BPMod.bp_IFB( GoodCoxCond, BPMod.bp_IFB( CoxDerog, (
CoxGood / DerogNotZero ) * 0.2, CoxGood * 0.2 ), 0 )
GoodCoxScaler = BPMod.bp_IFB( GoodCoxCond, BPMod.bp_MAX(
GoodCoxCredit * GoodCoxInc, 1 ) * GoodCoxInc, 0 )
BadBuyer = BPMod.bp_IFB( BPMod.bp_IFG( YrsTRW + Derog,10,1,0 ) *
BPMod.bp_IFG( CoxPoints,0,1,0 ), 1, 0 )
BadBuyerScaler = BPMod.bp_IFB( BadBuyer, BPMod.bp_MAX( 0,1 - 0.1 * (
YrsTrw + Derog - 10 ) ), 1 )
```

```
CoxScaler = BadBuyerScaler * ( CoxPoints + GoodCoxScaler * CoxPoints
)

'Note
'       CoxScaler is used in BeginFinalCFCalculation, MINTRW
'*********************************************************************
***



'*********************************************************************
***
'Calculate variable ResidTot for cust fact calc later
'*********************************************************************
***
'Input Parameters
' Resid

Resid8YearBase = BPMod.bp_IFGE( Resid, 8.1, BPMod.bp_MIN( Resid - 8,
4 ) * 0.00, 0 )
Resid5YearBase = BPMod.bp_IFGE( Resid, 5.1, BPMod.bp_MIN( Resid - 5,
3 ) * 1.44, 0 )
Resid1YearBase = BPMod.bp_IFGE( Resid, 1.1, BPMod.bp_MIN( Resid - 1,
4 ) * 1.27, 0 )
Resid0YearBase = BPMod.bp_IFGE( Resid, 0.0, BPMod.bp_MIN( Resid, 1 )
* 0.776, 0 )
ResidTot = Resid8YearBase + Resid5YearBase + Resid1YearBase +
Resid0YearBase - 0.176

'Note
'       ResidTot is used in FinalCFCalculation
'*********************************************************************
***


'*********************************************************************
***
'Calculate scaler for good/derog credit items="goodscaler",
"badscaler"
'Note GoodScaler/BadScaler could be done separately -- IMP
'*********************************************************************
***
'Input Parameters
' TotalDerog, TotalGood, RealHiGood, RealHiDerog, YrsTRW, vClass,
' BK, RealHiDerog

GSJustForPlaying = 1.50
GSHiGood = 0.25
GS2ManyAcct = - 0.25
GSGood2xDerog = 0.25
GSDerog2xGood = - 0.25
GSDerog5xGood = - 0.25
GSNoDerog = 0.20
GSFTB = 0.65
GSGoodMoreThanDerog = 0.10

GoodScalerBase = GSJustForPlaying
GoodScaler9 = BPMod.bp_IFE( TotalDerog,0, GoodScalerBase + GSNoDerog,
GoodScalerBase )
GoodScaler8 = BPMod.bp_IFG( TotalGood,TotalDerog, GoodScaler9 +
GSGoodMoreThanDerog, GoodScaler9 )
GoodScaler7 = BPMod.bp_IFB( BPMod.bp_IFG( RealHiGood,RealHiDerog *
10,1,0 ) * BPMod.bp_IFG( RealHiDerog,100,1,0 ) * BPMod.bp_IFL(
RealHiDerog,3000,1,0 ),GoodScaler8 + GSHiGood,GoodScaler8 )
GoodScaler6 = BPMod.bp_IFB( BPMod.bp_IFG( TotalGood,TotalDerog *
2,1,0 ) * BPMod.bp_IFGE( TotalDerog,1,1,0 ),GoodScaler7 +
GSGood2xDerog, GoodScaler7 )
GoodScaler5 = BPMod.bp_IFGE( TotalDerog,TotalGood * 2,GoodScaler6 +
GSDerog2xGood,GoodScaler6 )
GoodScaler4 = BPMod.bp_IFGE( TotalDerog,TotalGood * 5,GoodScaler5 +
GSDerog5xGood,GoodScaler5 )
```

```
GoodScaler3 = BPMod.bp_IFB( BPMod.bp_IFE( YrsTRW,0,1,0 ) *
BPMod.bp_IFNE( vClass,5,1,0 ),GoodScaler4 + GSFTB,GoodScaler4 )
GoodScaler2 = BPMod.bp_IFB( BPMod.bp_IFLE( YrsTrw,2,1,0 ) *
BPMod.bp_IFGE( TotalGood + TotalDerog,6,1,0 ),GoodScaler3 +
GS2ManyAcct,GoodScaler3 )
GoodScaler1 = BPMod.bp_IFL( RealHiDerog,1000,GoodScaler2 + ( 1000 -
RealHiDerog ) * 0.0005, GoodScaler2 )
GoodScaler0 = BPMod.bp_IFL( YrsTRW,1,GoodScaler1 + ( 1 - YrsTRW ) *
TotalGood * - 0.5, GoodScaler1 )
GoodScalerX = BPMod.bp_MIN ( GoodScaler0, 1.5 )
GoodScaler = BPMod.bp_MAX ( GoodScalerX, 0.25 )



BSSmallHD = - 0.20
BSBK = - 0.20
BadScalerBase = 1.05
BSHiDerog = 0.20

BadScaler5 = BPMod.bp_IFB( BPMod.bp_IFGE( RealHiDerog,5000,1,0 ) *
BPMod.bp_IFE( BK,0,1,0 ), BadScalerBase + BSHiDerog, BadScalerBase )
BadScaler4 = BPMod.bp_IFB( BK, BadScaler5 + BSBK, BadScaler5 )
BadScaler3 = BPMod.bp_IFLE( RealHiDerog, 500, BadScaler4 + BSSmallHD,
BadScaler4 )
'CHANGE
BadScaler2 = BPMod.bp_IFB( BPMod.bp_IFB( BK,1,0 ) * BPMod.bp_IFL(
YrsTRW,5,1,0 ), BadScaler3 + (5 - YrsTRW)*0.3, BadScaler3 )
BadScaler1 = BPMod.bp_MAX( BadScaler2, 1.00 )
BadScaler = BPMod.bp_MIN( BadScaler1, 1.5 )

'Note
'       GoodScaler is used in FinalCFCalculation
'       BadScaler is used in FinalCFCalculation
'*******************************************************************
***



'*******************************************************************
***
'Calculate BKBonus to be added to cust fact as part of finetune
'*******************************************************************
***
'Input Parameters
' vClass, TotalDown, Price, YrsTRW, TotalGood, RealHiGood, Spouse,
' Inc, SPInc, BK, minBK

BKStrong = 0.5
BKGood = 0.2
BKInc = 0.2
BKSpouse = 0.05
BKHiGood = 0.2

BKBonusCond = BPMod.bp_IFNE( vClass,5,1,0 ) * BPMod.bp_IFGE(
TotalDown,Price * 0.20,1,0 ) * BPMod.bp_IFGE( TotalDown,1500,1,0 ) *
BPMod.bp_IFGE( YrsTRW,5,1,0 ) * BPMod.bp_IFG( TotalGood,5,1,0 )
'this is according the dukes new expression
BKBonusExp6 = BPMod.bp_IFGE( RealHiGood,10000,BKHiGood,0 )
BKBonusExp5 = BPMod.bp_IFB( Spouse,BKBonusExp6 + BKSpouse,
BKBonusExp6 )
BKBonusExp4 = BPMod.bp_IFGE( BPMod.bp_IFB( Spouse,Inc + Spinc,Inc
),3000,BKBonusExp5 + BKInc, BKBonusExp5 )
BKBonusExp3 = BPMod.bp_IFGE( TotalGood,8,BKBonusExp4 + BKGood,
BKBonusExp4 )
BKBonusExp2 = BPMod.bp_IFE( MinBK,MinDiscStrongBK, BKBonusExp3 +
BKStrong, BKBonusExp3 )
BKBonusExp1 = BPMod.bp_IFB( BKBonusCond,BKBonusExp2, 0 )
BKBonus = BPMod.bp_MIN( BPMod.bp_IFB( BK, BKBonusExp1, 0 ), 1 )
'BKBonusCond = BPMod.bp_IFNE( vClass,5,1,0 ) * BPMod.bp_IFGE(
TotalDown,Price * 0.20,1,0 ) * BPMod.bp_IFGE( TotalDown,1500,1,0 ) *
BPMod.bp_IFGE( YrsTRW,5,1,0 ) * BPMod.bp_IFG( TotalGood,5,1,0 )
```

```
'Note
'       BKBonus is used in FinalCFCalculation
'*********************************************************************
***


'*********************************************************************
***
'Debt Model 1, Calculate countRent and crapRatio
'*********************************************************************
***
'Input Parameters
' Spouse, Debt, RealInc, Rent,

CRStart = 0.15
CRCountAll = 0.20

OKCrap = BPMod.bp_IFB( Spouse, 0.18, 0.13 )
Crap = DEBT / RealInc
'CHANGE
RentMult = ( Crap - CRStart ) / ( CRCountAll - CRStart )
CountRentExp2 = RentMult * Rent
CountRentExp1 = BPMod.bp_IFGE( Crap,CRCountAll,Rent,CountRentExp2 )
CountRent = BPMod.bp_IFG( Crap,CRStart,CountRentExp1,0 )
CrapRatio = BPMod.bp_MAX( Crap - OKCrap, 0 )

'Note
'       CountRent is used in TotalDebt
'       CrapRatio is used in PPAdjust
'*********************************************************************
***


'*********************************************************************
***
'Calculate SigDown
'*********************************************************************
***
'Input Parameters
' MaxCB, CB, WarrAllowance, RealDown, Price

SDDollarDown = 1500
SDPercentDown = 0.30
SDScaler = 0.80
SDEquityMult = 0.50

Equity = BPMod.bp_MAX(( MaxCB - CB - WarrAllowance ), 0 )
DollarDownMult = BPMod.bp_MIN( RealDown, SDDollarDown ) /
SDDollarDown
PercentDownMult = BPMod.bp_MIN( RealDown / Price, SDPercentDown ) /
SDPercentDown
SigMult = BPMod.bp_MAX( BPMod.bp_MAX( DollarDownMult, PercentDownMult
), SDEquityMult )
SigDown = BPMod.bp_MIN( SigMult * Equity * SDScaler, 0.5 * RealDown )

'Note SigDown is used in FTBBonus/SmallFTBBonus, HICBHIT, FineTune,
'       Excess Term Determination, DownPayment Probability, MINFact
'*********************************************************************
***


'*********************************************************************
***
'Calculate DebtAdjustment
'*********************************************************************
***
'Input Parameters
' Interest, CB, DaysToPay, TooSmallPmt, Term, Payment
```

```
DATerm = 30
DAScaler = 0.90

DAPmt = BPMod.bp_MAX( BPMod.bp_PMT(Interest, DATerm, CB, DaysToPay ),
TooSmallPmt )
DebtAdjustment = BPMod.bp_IFNE( Term,DATerm,( DAPmt - Payment ) *
DAScaler, 0 )

'Note
'       DebtAdjustment is used in TotalDebt
'*********************************************************************
***


'*********************************************************************
***
'Calculate TotalDebt
'*********************************************************************
***
'Input Parameters
' CountRent, RealInc, Debt, Ins, CB, Interest, Term, WarrAllowance,
DaysToPay
' Payment, DebtAdjustment

MinRent = 250

MinDebt = BPMod.bp_MAX( BPMod.bp_MAX( CountRent, MinRent ), RealInc *
0.1 ) + Debt
InsDebt = BPMod.bp_IFB( BPMod.bp_IFE( Ins,0,1,0 ) * BPMod.bp_IFG(
CB,2500,1,0 ), CB * 0.01, 0 )
WarDebtExp = BPMod.bp_PMT( Interest, Term, WarrAllowance, DaysToPay )
WarDebt = BPMod.bp_IFG( WarrAllowance,0, WarDebtExp,0 )
TotDebt = MinDebt + Payment + InsDebt - WarDebt + DebtAdjustment

'Note
'       TotDebt is used in RealJob, FinalCFCalculation, DebtProblem,
Error section
'*********************************************************************
***


'*********************************************************************
***
'Calculate Variable Time On Job whether married or not = "RealJob"
'*********************************************************************
***
'Input Parameters
' Job, Inc, Support, SpJob, SpInc, TotalInc, TotDebt, Spouse

JobInc = Job * ( Inc - Support )
SpJobInc = SpJob * SpInc
RealJobExp2 = ( JobInc + SpJobInc ) / TotalInc
'CHANGE
RealJobExp1 = BPMod.bp_IFLE( TotDebt / ( Inc - Support ),0.40,
BPMod.bp_MAX(Job, RealJobExp2), RealJobExp2 )
RealJob = BPMod.bp_IFB( Spouse, RealJobExp1, Job )

'Note
'       RealJob is used in JobTot, FTBBonus/SmallFTBBonus, DebtScaler
'*********************************************************************
***


'*********************************************************************
***
'Calculate JobTot to be used in Customer Factor Determination
'*********************************************************************
***
'Input Parameters
' RealJob
```

```
JobPoints1 = LookupJobTable( RealJob, 2 )
ExtraTime = RealJob - LookupJobTable ( RealJob, 1 )
JobPoints2 = LookupJobTable( RealJob, 3 ) * ExtraTime
JobTot = JobPoints1 + JobPoints2

'Note
'       JobTot is used in FinalCFCalculation
'*********************************************************************
***


'*********************************************************************
***
'Calculate Bonus Points For FTB or Short Bureau to be used as part of
'finetune = smallFTBBonus
'*********************************************************************
***
'Input Parameters
' YrsTRW, vClass, Repos, RealHiDerog, RealInc, Payment, CB, INS,
SigDown,
' PhBill, TotalDown, Price, Spouse, Resid, RealJob,

FTBINC = 1
FTBPMTRatio = 1
FTBCB = 1
FTBPhBill = 2
FTBDown = 0.20
FTBSpouse = 2
FTBResid = 3
FTBJob = 3

FTBPointsCond1 = BPMod.bp_IFLE( YrsTRW,1.1,1,0 ) * BPMod.bp_IFNE(
vClass,5,1,0 ) * BPMod.bp_IFE( Repos,0,1,0 ) * BPMod.bp_IFL(
RealHiDerog,3000,1,0 )
FTBPoints7 = BPMod.bp_IFGE( RealInc,1500,FTBInc,0 )
FTBPoints6 = BPMod.bp_IFLE( Payment / RealInc,0.20,FTBPoints7 +
FTBPmtRatio,FTBPoints7 )
FTBPoints5 = BPMod.bp_IFLE( CB - Ins - SigDown,5500,FTBPoints6 +
FTBCB,FTBPoints6 )
FTBPoints4 = BPMod.bp_IFB( PhBill,FTBPoints5 + FTBPhBill,FTBPoints5 )
FTBPoints3 = BPMod.bp_IFGE( (TotalDown / Price), 0.25, FTBPoints4 + 1
+ ( TotalDown / Price - 0.25 ) / FTBDown,FTBPoints4 )
FTBPoints2 = BPMod.bp_IFB( Spouse,FTBPoints3 + FTBSpouse,FTBPoints3 )
FTBPoints1 = BPMod.bp_IFGE( Resid,2.1,FTBPoints2 +
FTBResid,FTBPoints2 )
FTBPointsExp = BPMod.bp_IFGE( RealJob,2.1,FTBPoints1 +
FTBJob,FTBPoints1 )
FTBPoints = BPMod.bp_IFB( FTBPointsCond1, FTBPointsExp, 0 )
SmallFTBBonus = ( 1.1 - YrsTRW ) * 0.25 * BPMod.bp_MIN ( 1, FTBPoints
/ 6 )
FTBBonus = BPMod.bp_IFG( FTBPoints,6, ( 1.1 - YrsTRW ) * 0.50 *
BPMod.bp_MIN( 1, ( FTBPoints - 6 ) / 3 ), 0 )

'Note
'       FTBPoints used in MINTRW
'       SmallFTBBonus used in FineTune
'       FTBBonus is used in FineTune
'*********************************************************************
***


'*********************************************************************
***
'Begin Special Points Model; yields FTSpecialPoints
'Hit for low job and low resid at same time = shorttimehit
'*********************************************************************
***
'Input Parameters
' Resid, Job

STHit1 = 0.9
```

```
STHit2 = 0.6
STScaler1 = - 0.10
STScaler2 = - 0.20

ShortTimeHitCond1 = BPMod.bp_IFLE( Job,STHit1,1,0 ) * BPMod.bp_IFLE(
Resid,STHit1,1,0 ) * BPMod.bp_IFB( Spouse, BPMod.bp_IFLE(
SpJob,STHit1,1,0 ), 1 )
ShortTimeHitCond2 = BPMod.bp_IFLE( Job,STHit2,1,0 ) * BPMod.bp_IFLE(
Resid,STHit2,1,0 ) * BPMod.bp_IFB( Spouse, BPMod.bp_IFLE(
SpJob,STHit2,1,0 ), 1 )
ShortTimeHit1 = ( ( STHit1 * STHit1 ) - ( Job * Resid ) ) * STScaler1
ShortTimeHitExp1 = BPMod.bp_IFB( ShortTimeHitCond2, ShortTimeHit1 + (
( STHit2 * STHit2 ) - ( Job * Resid ) ) * STScaler2, ShortTimeHit1 )
ShortTimeHit = BPMod.bp_IFB( ShortTimeHitCond1, ShortTimeHitExp1, 0 )

'Note
'       ShortTimeHit is used in HICBHit, OptimalCBCredit
'*******************************************************************
***


'*******************************************************************
***
'Hit For Hi Amount Financed unless override=Y; = "HICBHIT"
'*******************************************************************
***
'Input Parameters
' CB, INS, SigDown, ShortTimeHit, Repos, BK, TotalGood, TotalDerog,
' RealHiGood, RealHiDerog, TotalLessIns,

HCBAmtFin = 8000
HCBScaler = - 0.00015

HiCBNumber = BPMod.bp_IFG( ( CB - Ins - SigDown ), HCBAmtFin, ( CB -
Ins - SigDown - HCBAmtFin ) * HCBScaler, 0 )
HCO1 = BPMod.bp_IFE( ShortTimeHit,0, 1, 0 )
HCO2 = BPMod.bp_IFE( Repos,0, HCO1 + 1, HCO1 )
HCO3 = BPMod.bp_IFAND2( BPMod.bp_IFE( Repos,1,1,0 ), BPMod.bp_IFB(
BK, 1,0 ), HCO2 + 1, HCO2 )
'HiCBOverideExp = BPMod.bp_IFGE( HCO3, 2, HCPExp, 1 )
'HiCBOveride = BPMod.bp_IFL( HiCBNumber, 0, HiCBOverideExp,1 )
'This is again corrected in mike dukes expression files.
HCP1 = BPMod.bp_IFGE( TotalGood, TotalDerog, 1, 0 )
HCP2 = BPMod.bp_IFGE( RealHiGood, RealHiDerog, HCP1 + 1, HCP1 )
HCP3 = BPMod.bp_IFGE( RealHiGood, 0.50 * ( TotalLessIns - SigDown ),
HCP2 + 1, HCP2 )
HCPExp = BPMod.bp_IFGE( HCP3, 2, 0, 1 )
HiCBOverideExp = BPMod.bp_IFGE( HCO3, 2, HCPExp, 1 )
HiCBOveride = BPMod.bp_IFL( HiCBNumber, 0, HiCBOverideExp,1 )
HiCBHit = HiCBNumber * HiCBOveride

'Note
'       HiCBHit is used in OptimalCBCredit
'*******************************************************************
***


'*******************************************************************
***
'Extra Points For Optimal CB = OptimalCBCredit
'*******************************************************************
***
'Input Parameters
' TotalLessIns, Payment, ShortTimeHit, RealDown, Variance, HiCBHit

OptimalCB = 5800
AllowVariance = 1700
OptimalPoints = 0.13

Variance = ABS( TotalLessIns - OptimalCB )
OptimalCBExp1 = BPMod.bp_IFB( BPMod.bp_IFGE( Payment, 240, 1,0 ) *
BPMod.bp_IFE( ShortTimeHit, 0, 1, 0 ) * BPMod.bp_IFGE( RealDown,
1000, 1, 0 ),( 1 - Variance / AllowVariance ) * OptimalPoints, 0 )
OptimalCBCredit = BPMod.bp_IFL( Variance, AllowVariance,
OptimalCBExp1, 0 )
FTSpecialPoints = OptimalCBCredit + HiCBHit + ShortTimeHit

'Note
'       FTSpecialPoints is used in FineTune
'********************************************************************
***


'********************************************************************
***
'FineTune Model to be added to customerfactor = "FineTune"
'********************************************************************
***
'Input Parameters
' FTBBonus, SmallFTBBonus, BKBonus, PhBill, TotalDerog, TotalGood,
SigDown
' EquityTest, TotalLessIns, MaxCB, Ins, RealHiDerog, YrsTRW, Repos,
BK, INSFlag

FTBonus = FTBBonus + SmallFTBBonus + BKBonus
FTPhBill = BPMod.bp_IFAND2( BPMod.bp_IFB( PhBill, 1, 0
),BPMod.bp_IFL( TotalDerog + TotalGood, 4, 1, 0 ), 0.12, 0 )
FTDerogHit = BPMod.bp_IFG( TotalDerog, 4, - 0.05 - 0.01 * (
TotalDerog - 5 ), 0 )
FTSigDown = SigDown * .0001 + BPMod.bp_IFG( SigDown, 2000, ( SigDown
- 2000 ) * .0001, 0 )
FTEquity = 0.75 - EquityTest + BPMod.bp_MAX( 0.6 - EquityTest, 0 )
FTBuyIFBreathing = BPMod.bp_MAX( FTSigDown, FTEquity ) *
BPMod.bp_IFOR2( BPMod.bp_IFL( TotalLessIns, ( MaxCB - Ins ) * 0.75,
1, 0 ), BPMod.bp_IFGE( SigDown, 1000, 1, 0 ), 1, 0 )
FTSmallHiDerog = BPMod.bp_IFB( BPMod.bp_IFE( TotalGood, 0, 1, 0 ) *
BPMod.bp_IFLE( RealHiDerog, 500, 1, 0 ) * BPMod.bp_IFG( YrsTRW, 1, 1,
0 ), 0.30, 0 )
FTBHD = BPMod.bp_IFB( BPMod.bp_IFGE( RealHiDerog, 2700, 1, 0 ) *
BPMod.bp_IFB( BK, 0, 1 ) * BPMod.bp_IFE ( Repos, 0, 1, 0 ),(
RealHiDerog / 8000 ) * - 0.60, 0 )
FTBigHiDerog = BPMod.bp_MAX( - 0.50,FTBHD )
InsCantFindErr = BPMod.bp_IFB( InsFlag, LookupIns( TotalLessIns,3 ),
0 )
FineTune = FTSpecialPoints + FTBigHiDerog + FTSmallHiDerog +
FTBuyIFBreathing + FTDerogHit + FTPhBill + FTBonus + InsCantFindErr

'Note
' FineTune is used in FinalCFCalculation
'********************************************************************
***


'********************************************************************
***
'Begin Final Customer Factor Calculation -- Add Up "F" Variables
'********************************************************************
***
'Input Parameters
' YrsTRW, JobTot, ResidTot, TotalGood, RealHiGood, TotalDerog, Bk,
TotalLessIns,
' EquityTest, RealHiDerog, BK, Home, RealInc, SpJob, SpGood, Spouse,
Cox, FineTune


CFTRWScaler = 0.75
CFJobScaler = 0.90
CFResidScaler = 0.60
CFHiGoodScaler = 0.90
CFBKScaler = 1.00
CFHomeScaler = 0.80
CFIncScaler = 0.075
```

```
CFDebtScaler = 1.00

TRWPart = BPMod.bp_IFL ( YrsTRW, 2, BPMod.bp_MIN( YrsTRW * 0.5, 0.9
), BPMod.bp_MIN( 0.7 + YrsTRW * 0.1, 1 ) )
FTRW = TRWPart * CFTRWScaler

JobPart = JobTot / 10
FJob = JobPart * CFJobScaler

ResidPart = ResidTot / 10
FResid = ResidPart * CFResidScaler

GoodPart = BPMod.bp_IFL ( TotalGood, 2, TotalGood * 0.5,
BPMod.bp_MIN( 0.5 + TotalGood * 0.1, 1 ) )
FGood = GoodPart * GoodScaler

HiGoodPart = BPMod.bp_IFL ( RealHiGood, 20000, 0.5 * RealHiGood /
20000, 0.5 )
FHiGood = HiGoodPart * CFHiGoodScaler

DerogPart = BPMod.bp_IFL ( TotalDerog, 4, TotalDerog * - 0.25, - 0.5
- TotalDerog * 0.1 )
BKDerog = BPMod.bp_IFB ( BK, 0.7, 1 )
FDerog = BPMod.bp_MAX( DerogPart * BadScaler, - 1.05 ) * BKDerog

CFPhBillScaler = BPMod.bp_IFL( TotalLessIns, 4000, 0.8, 0.65 ) * 20 /
Term
PhBillPart = BPMod.bp_IFB ( PhBill, BPMod.bp_IFL ( EquityTest, 0.90,
0.33, 0.33 * 0.80 ), 0 )
FPhBill = PhBillPart * CFPhBillScaler


RepoPart = Repos * - 0.25
CFRepoScaler = BPMod.bp_IFG( RealHiDerog, 1000, 2, BPMod.bp_MAX( 1,
RealHiDerog * .002 ) )
FRepo = RepoPart * CFRepoScaler

BKPart = BPMod.bp_IFB( BK, - 0.5, 0 )
FBK = BKPart * CFBKScaler

'CHANGE
HomePart = BPMod.bp_IFB ( Home, 2/3, 0 )
'CHANGE--ADD THIS
HomePartScaler= 0.4 + 0.4*(BPMod.bp_IFG (RealHiGood, 30000,
RealHiGood-30000, 0)/70000)
FHome = HomePart * BPMod.bp_MIN( CFHomeScaler, HomePartScaler)


IncPart = BPMod.bp_IFL ( RealInc, 3000, RealInc / 2000, BPMod.bp_MIN(
RealInc,12000 ) / 1800 )
FInc = IncPart * CFIncScaler

DebtPart = BPMod.bp_IFGE ( TotDebt / RealInc, 0.55, - 0.1,
BPMod.bp_MIN( 0.7 - TotDebt / RealInc, 0.5 ) )
FDebt = DebtPart * CFDebtScaler

CFSpouseScaler = BPMod.bp_IFLE( YrsTRW, 1, 0.5, 0.35 )
WorthlessSpouse = BPMod.bp_IFAND2( BPMod.bp_IFLE( SpJob, 0, 1, 0
),BPMod.bp_IFLE( SpGood, 0, 1, 0 ),0, 1 )
SpousePart = BPMod.bp_IFB ( Spouse, 0.5, 0 ) * WorthlessSpouse
FSpouse = SpousePart * CFSpouseScaler


CoxPart = BPMod.bp_IFB ( Cox, 0.5, 0 )
CFCoxScaler = CoxScaler / 10
FCox = CoxPart * CFCoxScaler

TotalCFPoints = FTRW + FJob + FResid + FGood + FHiGood + FDerog +
FPhBill + FRepo + FBK + FHome + FInc + FDebt + FSpouse + FCox +
FineTune
CustFact = BpMod.bp_ROUND(BPMod.bp_MAX( BPMod.bp_MIN(TotalCFPoints, 5
), 0.001 ) * 0.98, 2)


'Note
'       DebtPart is used in Error Section
'       FCox is used in Error Section
'       CustFact is used in ExcessTerm Determination, XTerm,
CFComponent =
'                   CFAllowance, DownPayment Probability, MinFact,
KinKTerm,
'                   FinalReserve, Error Section
'*******************************************************************
***
'Customer Factor Calculation ends here
'*******************************************************************
***




'*******************************************************************
***
'Calculate Scaler if good customer with high debt = "debtscaler"
'*******************************************************************
***
'Input Parameters
' TotalLessIns, RealInc, RealJob, YrsTRW, TotalDerog, RealHiDerog,
TotalGood

DebtScaler_exp1 = BPMod.bp_IFOR2( BPMod.bp_IFLE( TotalLessIns,RealInc
* 5,1,0 ),BPMod.bp_IFLE( TotalLessIns,4500,1,0 ),1,0 )
DebtScaler_exp2 = BPMod.bp_IFGE( RealJob,1,1,0 )
DebtScaler_exp3 = BPMod.bp_IFGE( YrsTRW,1,1,0 )
DebtScaler_exp4 = BPMod.bp_IFOR2( BPMod.bp_IFLE( TotalDerog,1,1,0
),BPMod.bp_IFLE( RealHiDerog,400,1,0 ),1,0 )
DebtScaler_exp5 = BPMod.bp_IFLE( TotalGood,4,1,0 )
DebtScaler_exp6 = BPMod.bp_IFL( TotalGood,TotalDerog,1,0 )
DebtScaler_exp7 = BPMod.bp_IFL( RealInc,1700,1,0 )
DebtScalerCondition = DebtScaler_exp1 * DebtScaler_exp2 *
DebtScaler_exp3 * DebtScaler_exp4 * DebtScaler_exp5 * DebtScaler_exp6
* DebtScaler_exp7

DSInc = BPMod.bp_MAX( RealInc, 1200 )
DebScalerExp = 0.5 + ( DSInc - 1200 ) / 1000
DebtScaler = BPMod.bp_IFB( DebtScalerCondition,DebScalerExp,1 )

'Note
'       DebtScaler is used in DebtProblem, PPAdjust
'*******************************************************************
***


'*******************************************************************
***
'Calculate debt ratio hit for pay prob adjustments = "debtproblem"
'*******************************************************************
***
'Input Parameters
'RealInc, TotDebt, EquityTest, DebtScaler

DebtRatio = RealInc / TotDebt
DebtHisExp = BPMod.bp_IFLE( DebtRatio,2, 0.225 + ( 2 - DebtRatio ) *
0.6, ( 2.5 - DebtRatio ) * 0.45 )
DebtHit = BPMod.bp_IFLE( DebtRatio, 2.5, DebtHisExp,0 )
DHMax1 = BPMod.bp_MAX( 0.95 - EquityTest, 0 )
DHMax2 = BPMod.bp_MAX( 0.75 - EquityTest, 0 )
DebtHitScaler = 1.05 - DHMax1 - DHMax2
DebtProblem = DebtHit * DebtHitScaler * DebtScaler
```

```
'Note
'       DebtProblem is used in PPAdjust
'*********************************************************************
***


'*********************************************************************
***
'Excess Term Determination Model Beyond baseterm
'Models yield "freeterm", "buyterm", & "exterm"
'*********************************************************************
***
'Input Parameters
' SigDown, Payment, CustFact, vClass, Miles, CarAge, CB, INS, Ins

FreeGetNone = 1.75
FreeGetAll = 2.30
SBGetNone = 2.75
SBGetAll = 3.25
BaseTerm = 31

MinPmt = 255 - ( SigDown / 75 )
OKPmt = BPMod.bp_IFGE( Payment,MinPmt,1,0 )

RegularFreeTerm = BPMod.bp_IFG( CustFact, FreeGetNone, 1, 0 )

YEMiles = LookupTermTable( vClass, 2 )
YEAge = LookupTermTable( vClass, 3 )
MEAge = LookupTermTable( vClass, 4 )
MEMiles = LookupTermTable( vClass, 5 )

FreeTermPercent = BPMod.bp_MIN( ( CustFact - FreeGetNone ) / (
FreeGetAll - FreeGetNone ), 1 )
Term4NewerCar = BPMod.bp_IFLE( Miles, YEMiles, BPMod.bp_MAX( YEAge -
CarAge, 0 ), 0 ) * FreeTermPercent
Term4LowMiCar = BPMod.bp_IFLE( CarAge, MEAge, BPMod.bp_MAX( ( MEMiles
- Miles ) / 5000, 0 ), 0 ) * FreeTermPercent
StrongBuyerFreeTerm = BPMod.bp_IFG ( CustFact, SBGetNone, 1, 0 )

SBAge = LookupTermTable( vClass, 6 )
SBMiles = LookupTermTable( vClass, 7 )


SBFreeTermPercent = BPMod.bp_MIN( ( CustFact - SBGetNone ) / (
SBGetAll - SBGetNone ), 1 )
Term4StrongBuyer = BPMod.bp_IFAND2( BPMod.bp_IFLE ( CarAge, SBAge, 1,
0 ),BPMod.bp_IFLE ( Miles, SBMiles, 1, 0 ),3 * SBFreeTermPercent, 0 )
QualifyFreeTerm = BPMod.bp_IFAND2( BPMod.bp_IFB( RegularFreeTerm, 1,
0 ),BPMod.bp_IFB( StrongBuyerFreeTerm, 1, 0 ),Term4NewerCar +
Term4LowMiCar + Term4StrongBuyer,BPMod.bp_IFB ( RegularFreeTerm,
Term4NewerCar + Term4LowMiCar, 0 ) )
FreeTerm = BPMod.bp_IFG( Term, BaseTerm, BPMod.bp_MIN(
QualifyFreeTerm, Term - BaseTerm ), 0 ) * OKPmt

OKTerm = BaseTerm + FreeTerm
BuyTerm = BPMod.bp_MAX( Term - OKTerm, 0 )

ExTermScaler = ( CustFact - 1 ) / 1.75 * 0.01

ExcessCharge = BPMod.bp_IFG ( CustFact, 1, 0.015 - ExTermScaler,
0.015 )
CostPerMonth = BPMod.bp_MAX( ExcessCharge, .005 )
PmtBelow250 = BPMod.bp_IFL ( Payment, 250, 1000, 1 )
TooLong = BPMod.bp_IFG ( BuyTerm, 6, 1000, 1 )
MustBuyTerm = BPMod.bp_IFGE ( BuyTerm, 0, 1, 0 )
ExTerm = BPMod.bp_IFB( MustBuyTerm, CostPerMonth * BuyTerm *
PmtBelow250 * TooLong, 0 ) * ( CB - Ins - WarrAllowance )
'ExTerm Determination Ends Here

'Note
'       FreeTerm is used in Xterm
'       BuyTerm is used in Xterm
'       ExTerm is used in Down Payment Probability, SpreadNum, Final
Reserve,
'               Error Section
'*********************************************************************
***


'*********************************************************************
***
'Primary term hit/helper = xterm
'*********************************************************************
***
'Input Parameters
' CustFact, CarAge, vClass, MaxCB, Ins, Miles, Term

TermCust = CustFact * 20
KentTerm = ( 12 - CarAge ) * 6
ClassTerm = 5 - vClass
ClassScaler = ClassTerm / 5

CBTerm = BPMod.bp_IFG ( ( MaxCB - Ins ), 6000, ( MaxCB - Ins - 6000 )
/ 500, 0 )
TermCFScaler = BPMod.bp_IFG ( CustFact, 1, BPMod.bp_MIN( CustFact -
1, 1 ), 0 )
TermCar = KentTerm + ( ClassTerm + CBTerm * ClassScaler ) *
TermCFScaler
TermMaxMiles = 180000 - ( vClass * 10000 )
SubtractTerm = BPMod.bp_IFG( Miles, TermMaxMiles, ( ( Miles -
TermMaxMiles ) / 10000 ) * vClass / 2, 0 )
TermMax = BPMod.bp_MIN( TermCar, TermCust ) + BuyTerm * 0.5 +
FreeTerm * 0.5 - SubtractTerm
XTerm = Term - TermMax

'Note
'       XTerm is used in PPAdjust
'*********************************************************************
***

'OK Till Here


'*********************************************************************
***
'Calculate InputDiscount
'Note separated the input discount to calculate from down payment
probability
'as it used in other places
'*********************************************************************
***
'Input Parameters
' CustFact, Reserve

FedExTax = BPMod.bp_IFGE( CustFact, 2.5, 0, BPMod.bp_MIN( ( 2.5 -
CustFact ) * 76, 39 ) )
InputDiscount = Reserve - FedExTax

'Note
'       InputDiscount is used in downpayment probability, spreadnum,
Error Section
'*********************************************************************
***


'*********************************************************************
***
'                                   PAYMENT PROBABILITY MODEL
'*********************************************************************
***
'*********************************************************************
***
```

```
'CUSTOMER FACTOR COMPONENT = "CFALLOWANCE"
'*********************************************************************
***
'Input Parameters
' CustFact

CFSMin = LookupCFScalerTable( CustFact, 1 )
CFSBase = LookupCFScalerTable( CustFact, 2 )
CFSExtra = LookupCFScalerTable( CustFact, 3 )
CustFactScaler = CFSBase + ( CustFact - CFSMin ) * CFSExtra
CFAllowance = CustFactScaler * CustFact

'Note
'       CFAllowance used in calculating PayProb
'*********************************************************************
***

'*********************************************************************
***
'Down Payment Probability="DownPrice"
'*********************************************************************
***
'Input Parameters
' Price, InputDiscount, SigDown, ExTerm

'FedExTax = BPMod.bp_IFGE( CustFact, 2.5, 0, BPMod.bp_MIN( ( 2.5 -
CustFact ) * 76, 39 ) )
'InputDiscount = Reserve - FedExTax
DownAllowance = ( Price * 0.2 ) + InputDiscount + SigDown - ExTerm
DownPrice = DownAllowance / Price

'Note
'       DownPrice usedin calculating PayProb
'*********************************************************************
***


'*********************************************************************
***
'OVERALL SCALER
'*********************************************************************
***
PPScaler = 0.95

'Note
'       PPScaler is used in calculating PayProb
'*********************************************************************
***


'*********************************************************************
***
'ADJUSTMENTS = "PPADJUST"
'*********************************************************************
***
'Input Parameters
' DebtProblem, CrapRatio, DebtScaler, Payment, Term, Xterm

StupidNum = 8
StupidTerm = 17

PPDebt = DebtProblem * - 0.7
PPCrap = ( CrapRatio * DebtScaler ) * - 1
PPStupid = BPMod.bp_IFB( BPMod.bp_IFL( Payment / Term,StupidNum,1,0 )
* BPMod.bp_IFGE( Term,StupidTerm,1,0 ), ( StupidNum - Payment / Term
) * - 0.1, 0 )
PPTerm = XTerm * - 0.01
PPAdjust = PPTerm + PPDebt + PPStupid + PPCrap

'Note
'       PPStupid is used in Error Section
```

20507954 " 042502

```
'       PPTerm is used in Error Section
'       PPAdjust is used to calculate PayProb
'*********************************************************************
***

PayProb = CFAllowance * DownPrice * PPScaler + PPAdjust
'Note
'       PayProb is used in SpreadNum
'*********************************************************************
***
'                                                End Payment Probability
'*********************************************************************
***


'*********************************************************************
***
'DISCOUNT NEEDED BASED ON PAYMENT PROBABILITY MODEL = "SPREADNUM"
'*********************************************************************
***
'Input Parameters
' PayProb, InputDiscount, TotalLessIns, WarrAllowance, ExTerm,
DiscountAllow
' CB, Ins, WarrAllowance

SpreadNumScaler = 0.50
SpreadReq = 0.12

LossProb = BPMod.bp_MIN ( 1 - PayProb, 1.1 )
DiscountAllow = InputDiscount * 2
LossAmount = LossProb * ( TotalLessIns - WarrAllowance ) + ExTerm -
DiscountAllow
Spread = SpreadReq * ( CB - Ins - WarrAllowance )
SpreadNum = ( LossAmount + Spread ) * SpreadNumScaler

'Note
'       SpreadNum is used in Final Reserve
'*********************************************************************
***


'*********************************************************************
***
'MINIMUM % DISCOUNT AREA
'CALCULATE MIN % DISCOUNT DEPENDING OF # REPOS = "MINREPO"
'*********************************************************************
***
'Input Parameters
' Repos, BK

MinRepoExp3 = BPMod.bp_CASE3( repos, 1, 2, 3, 0.125, 0.20, 0.35, 0.50
)
MinRepoExp2 = BPMod.bp_CASE2( repos, 1, 2, 0.10, 0.175, 0.30 )
MinRepoExp1 = BPMod.bp_IFB( BK,MinRepoExp2, MinRepoExp3 )
MinRepo = BPMod.bp_IFE( Repos,0,0.10,MinRepoExp1 )

'Note
'       MinRepo is used in FinalReserve
'*********************************************************************
***


'*********************************************************************
***
'CALCULATE MIN % DISCOUNT BASED ON HI DEROG = "MINDEROG"
'*********************************************************************
***
'Input Parameters
' RealHiDerog, BK, MinDiscount

MinDiscHiDerog = 0.12
```

```
MinDerog = BPMod.bp_IFB( BPMod.bp_IFGE( RealHiDerog,3000,1,0 ) *
BPMod.bp_IFE( BK,0, 1,0 ), MinDiscHiDerog, MinDiscount )

'Note
'       MinDerog is used in Final Reserve
'*********************************************************************
***


'*********************************************************************
***
'CALCULATE MIN % DISCOUNT BASED BK=YES AND OTHER FACTORS = "MINBK"
'*********************************************************************
***
'Input Parameters
' TotalDown, RealInc, TotalGood, Home, Spouse, RealHiGood, BK,
YrsTRW, vClass
' MinDiscount

MBKDown = 1
MBKInc = 3
MBKHome = 1
MBKSpouse = 1
MBKMinPoints = 6
MinDiscStrongBK = 0.11
MinDiscRegularBK = 0.15
MBKGood = 3
MBKHiGood = 3

BKPoints6 = BPMod.bp_IFGE( TotalDown,3000,MBKDown,0 )
BKPoints5 = BPMod.bp_IFGE( RealInc,3000,BKPoints6 + MBKInc,BKPoints6
)
BKPoints4 = BPMod.bp_IFGE( TotalGood,8,BKPoints5 + MBKGood,BKPoints5
)
BKPoints3 = BPMod.bp_IFB( Home,BKPoints4 + MBKHome, BKPoints4 )
BKPoints2 = BPMod.bp_IFB( Spouse,BKPoints3 + MBKSpouse,BKPoints3 )
BKPoints1 = BPMod.bp_IFGE( RealHiGood,10000, BKPoints2 +
MBKHiGood,BKPoints2 )
MinBKExp2 = BPMod.bp_IFGE( BKPoints1, MBKMinPoints, MinDiscStrongBK,
MinDiscRegularBK )
MinBKCon = BPMod.bp_IFB( BK,1,0 ) * BPMod.bp_IFGE( RealInc,2400,1,0 )
* BPMod.bp_IFGE( TotalGood,5,1,0 ) * BPMod.bp_IFGE( YrsTRW,8,1,0 ) *
BPMod.bp_IFNE( vClass,5,1,0 )
MinBKExp1 = BPMod.bp_IFB( MinBKCon, MinBKExp2,MinDiscRegularBK )
MinBK = BPMod.bp_IFB( BK,MinBKExp1,MinDiscount )

'Note
'       MinBK is used in FinalReserve and BKBonus
'       BKBonus is defined above so that should be redefined or this
needs to
'       move up there somewhere
'*********************************************************************
***


'*********************************************************************
***
'CALCULATE MIN % DISCOUNT BASED ON LOW TIME ON BUREAU = "MINTRW"
'*********************************************************************
***
'Input Parameters
' FTBPoints, CoxScaler, YrsTRW

MinTRWExp = BPMod.bp_IFOR2( BPMod.bp_IFGE( FTBPoints,9,1,0 ),
BPMod.bp_IFGE( CoxScaler,30,1,0 ), 0.125 - ( YrsTRW / 40 ), 0.15 - (
YrsTRW / 20 ) )
MinTRW = BPMod.bp_IFL( YrsTRW, 1, MinTRWExp , 0.10 )

'Note
'       MinTRW is used in Final Reserve
```

```
'*********************************************************************
***


'*********************************************************************
***
'CALCULATE MIN % DISCOUNT BASED ON CustFact= "MINFact"
'*********************************************************************
***
'Input Parameters
' SigDown, TotalLessIns, WarrAllowance, CustFact, MinDiscount

FactMinDisc = 0.3

SigDownHelper = ( SigDown * 0.25 ) / ( TotalLessIns - WarrAllowance )
Below75 = BPMod.bp_IFL( CustFact, 0.75, 1, 0 )
Below75Hit = BPMod.bp_IFL( CustFact, 0.35, .2, ( 75 - ( CustFact *
100 ) ) * .005 )
Below35 = BPMod.bp_IFL( CustFact, 0.35, 1, 0 )
Below20 = BPMod.bp_IFL( CustFact, 0.20, 1, 0 )
LowBalScaler = BPMod.bp_IFLE( TotalLessIns, 2000, 0.50,
BPMod.bp_IFLE( TotalLessIns, 3000, 1 - ( ( 3000 - TotalLessIns ) /
1000 ) * 0.50, 1 ) )
MinFact75 = ( FactMinDisc + Below75Hit - SigDownHelper ) *
LowBalScaler
MinFact35 = BPMod.bp_IFB( Below35, BPMod.bp_IFB( Below20, 10,
BPMod.bp_IFG( TotalLessIns, 3000, 10, 0 ) ), 0 )
MinFact = BPMod.bp_IFB( Below75, BPMod.bp_MAX( MinFact75, MinFact35
), MinDiscount )

'Note
'       MinFact is used in FinalReserve
'*********************************************************************
***


'*********************************************************************
***
'ADDITIONAL DISCOUNT FOR KINKY TERM = "KINKTERM"
'*********************************************************************
***
'Input Parameters
' Term, CarAge, Miles, CustFact

CostPerKinkPoint = 2
KinkAge = 8
KinkMiles = 120000
KinkCF = 1.70
KinkMaxTerm = 28

TermIsKinky = BPMod.bp_IFG( Term, KinkMaxTerm, 1, 0 )
KinkSubtot = BPMod.bp_MAX( CarAge - KinkAge, 0 ) + BPMod.bp_MAX(
Miles - KinkMiles, 0 ) / 10000
PointsFromCF = BPMod.bp_MAX( KinkCF - CustFact, 0 ) * 10 * KinkSubTot
OverMax = BPMod.bp_MIN( BPMod.bp_MAX( Term - KinkMaxTerm, 0 ), 3 )
TotalKinkPoints = ( KinkSubtot * OverMax ) + ( KinkSubtot *
PointsFromCF * OverMax )
KinkTerm = BPMod.bp_IFB( TermIsKinky, TotalKinkPoints *
CostPerKinkPoint, 0 )

'Note
'       KinkTerm is used in Final Reserve, Error Section
'*********************************************************************
***


'*********************************************************************
***
'Get Final Reserve
```

```
'*********************************************************************
***
'Input Parameters
' CustFact, MinDerog, MinTrw, MinBK, MinRepo, MinFact, MinDiscount,
' TotalLessIns, WarrAllowance, MinDisc, SpreadNum, Term, Payment

MinDisc = BPMod.bp_IFGE( CustFact, 2.5, 300, BPMod.bp_MIN( ( 2.5 -
CustFact ) * 88 + 300, 344 ) )
MinPercent = BPMod.bp_MAX( BPMod.bp_MAX( BPMod.bp_MAX( MinDerog,
MinTRW ), BPMod.bp_MAX( MinBK, MinRepo ) ), BPMod.bp_MAX( MinFact,
MinDiscount ) )
MinReserve = MinPercent * ( TotalLessIns - WarrAllowance )
FinalSubtot = BPMod.bp_MAX( BPMod.bp_MAX( MinDisc, MinReserve ),
SpreadNum )
TooMuchTerm = BPMod.bp_IFG( Term, 48, 50000, 0 )
PmtTooSmall = BPMod.bp_IFL( Payment, TooSmallPmt, 50000, 0 )

FinalReserve = FinalSubtot + KinkTerm + ExTerm + TooMuchTerm +
PmtTooSmall
'Note
'       MinDisc is used in Error Section
'       MinReserve is used in Error Section
'       FinalReserve is used Error Section and StrucOk var
'*********************************************************************
***


'*********************************************************************
***
'GET OVERADVANCE AND CHECK TO DEALERs
'*********************************************************************
***
'Input Parameters
' CB, MaxCB, INS, Reserve, ACQFEE

REALOA=BPMod.bp_IFG(CB, MAXCB, CB-MAXCB, 0.00)
CheckToDealer=CB-INS-RESERVE-ACQFEE-REALOA
OA=Round(REALOA+0.50, 0)
'Note
'       RealOA is used in Error Section
'       CheckToDealer is an output
'       OA is used in Error Section
'*********************************************************************
***


'Taken care when we found realinc
'if RealInc <= 0 then
'   RealInc = 1
'end if


'*********************************************************************
***
'HINT AND ERROR SECTION
'NEED THE FOLLOWING TO BEGIN HINTS
'*********************************************************************
***
DebtP= TotDebt/RealInc
DebtDiff= DebtP - 0.55
LessDebt= DebtDiff*RealInc + 5
GetDown= (2000-RealDown)*0.8
LowerPrice= (1000-SigDown-GetDown)/0.8


'CHANGE
if repos > 3 then
hint1 = " Wow! " + formatnumber(repos,0) + " repossessions!!!  "
end if
```

```
'CHANGE
if repos > 2 then
hint2 = "  But...  " + formatnumber(repos,0) + " repossessions?
Forget the phone bill, get a blood sample.  "
end if

'CHANGE
if ((PPStupid+PPTerm < -0.15) and (FinalReserve > (CB-Ins)*0.15) and
(FinalReserve > 500)) then
hint3 = " You could do better with a shorter term.  "
end if

'CHANGE
if ((DealerGross < 0) and (RealInc < 1400)) then
hint4= " Try a less expensive car for this income so you can make a
better deal."
end if

'CHANGE
if ((CustFact < 0.75) and (MinRepo*(TotalLessIns-WarrAllowance) <=
FinalReserve-200)) then
hint5= " Try a lower price, or more down, or a shorter term.  You
might make a better deal.  "
end if

'CHANGE
if ((YrsTRW = 0) and (Good > 0) and (Good < 3)) then
hint6= " Make sure you get documentation showing the good credit.  No
rental, medical, or dental. "
end if

'CHANGE
if ((Home = 1) and (HiGood < 30000)) then
hint7= " If the house is not on the credit bureau, then make sure to
send proof of home-owner.  "
end if

'CHANGE
'if ((Miles <> 117545) or (Price <> 6995)) then
if (InputDiscount >= FinalReserve) then
hint8= " It's a deal!   "
end if

'CHANGE
if (Miles < 100000 and (BPMod.bp_THISYEAR-CarYear > 9)) then
hint9= " Better check the miles.  If the car is over 10 years old,
you have to input at least 100,000 miles."
end if

'CHANGE
if Miles < (BPMod.bp_THISYEAR-CarYear)*7000 then
hint10= " Check your miles.  Your input is very low, unless the last
owner was my grandmother."
end if

'CHANGE
if repos >= 5 then
hint11= "  Like a '72 Pinto.  "
end if

'CHANGE
if ((Repos > 0) and (HiDerog < 3000)) then
hint12= " Don't forget that the Hi Derog is the amount of the loan,
not how much was charged off.  "
end if

'CHANGE
if BK = 1 then
hint13= " Bankruptcy must be discharged.  "
end if
```

```
'CHANGE
if ((YrsTRW = 0) and (Good > 2)) then
hint14= " You can't have more than 2 good credit items that are not
on the bureau.  "
end if

'CHANGE
if ((Job > 2) and (Resid > 2) and (Job = Resid)) then
hint15= " If this is a military deal, don't forget to send a
completed Mac allotment.  Must be rank of E3 or higher.  "
end if

'CHANGE
if Support > 0 then
hint16= "  Remember not to count Family Support accounts as Good or
Derog. "
end if

'CHANGE
if ((DebtPart < 0) and (LessDebt < 40) and (FinalReserve >=
BPMod.bp_MAX(MinDisc, MinReserve) + KinkTerm + ExTerm + 100) and
(Payment-LessDebt > 170)) then
hint17= " You can make a better deal if you use Price and Down to get
the payment about " + formatnumber(LessDebt,0) + " dollars lower.  "
end if

'CHANGE
if ((DebtPart < 0) and (LessDebt > 40) and (FinalReserve >=
BPMod.bp_MAX(MinDisc, MinReserve) + KinkTerm + ExTerm + 100) and
(Payment-LessDebt > 170)) then
hint18= "  You could make a lot better deal if the payment was " +
formatnumber(LessDebt,0) + " dollars lower.  Try a less expensive
car.  "
end if

'CHANGE
if GetDown+SigDown <= 1000 then
hint19= " And lower the Price by about " + formatnumber(LowerPrice,
0) + " dollars. "
end if

'CHANGE
if ((SigDown >= 850) and (SigDown < 1000) and (FinalReserve >=
BPMod.bp_MAX(MinDisc, MinReserve) + KinkTerm + ExTerm + 100)) then
  if RealDown < 2000 then
    hint20= "  You might do better if you get 2000 dollars down. "
+hint19
  else
    hint20= "  You might do better if you lower the Price by about "
+ formatnumber((1000-SigDown)/0.8, 0) + " dollars.  "
  end if
end if

'CHANGE
hint22= " Try putting down  " + dollarString(OA,0) + "  more, or
lower the price."

if ((CustFact > 1.0000000000) and (OA > 0)) then
  hint21= " Try putting down  " + formatnumber(OA,0) + " dollars
more, or reserve the O-A, then:"
  else if (OA > 0) then
    hint21= ""
    hint25=hint22
    hint8= ""
  end if
end if

if (CustFact < 1.000000000) then
  hint23= " You can't reserve the O-A, because the Customer Factor
has to be over 1. "
else
```

```
  hint23= ""
end if

'if (vClass = 5) then
'  hint24= " You can't reserve the O-A, because the Car Class cannot
be 5. "
'else
'  hint24= ""
'end if


'[ERROR CHECKING]
DeathErr4 = BPMod.bp_IFG( KinkTerm,( CB - Ins ) * .1, 4, 0 )
DeathErr3 = BPMod.bp_IFL( Payment, TooSmallPmt, 3, DeathErr4 )
DeathErr2 = BPMod.bp_IFGE( ExTerm, 0.25 * ( CB - Ins ), 2, DeathErr3
)
DeathErr1 = BPMod.bp_IFG( Term, 48, 1, DeathErr2 )

Err9 = BPMod.bp_IFL( FCox,0,9,10 )
Err8 = BPMod.bp_IFG( KinkTerm,( CB - Ins ) * .02, 8, Err9 )
'Err7 = BPMod.bp_IFG( CB, MAXCB, 7, Err8 )

if (CB-MAXCB <= 0.00) then
  Err7 = Err8
  else
  if (CB-MAXCB < 300) then
    Err7 = 12
    else
    if (CB-MAXCB <= 1000) then ' [o/a is between 300 and 1000,
inclusive]
      Err7 = 7
      else
      Err7 = 11
    end if
  end if
end if

Err6 = BPMod.bp_IFL( Reserve, 300, 6, Err7 )
Err5 = BPMod.bp_IFL( Reserve, .10 * ( CB - Ins - WarrAllowance ), 5,
Err6 )


ErrCode = BPMod.bp_IFG( FinalReserve, CB, DeathErr1, Err5 )
Errstr = ErrLookup(ErrCode)


'---------------------------------Added for Stand Alone
BP---------------------------------
Errstr = ErrLookup(ErrCode)
NoDollarOA=FormatNumber(REALOA,0)

if (REALOA = 0.00) then
  OAStr = ""
else
  OAStr = "$ " & FormatNumber(OA,0)
end if



'*********************************************************************
***
'Structure OK and Amount OK
'*********************************************************************
***
'Input Parameters
' InputDiscount, FinalReserve, CB, MAXCB

StructOK  = InputDiscount >= FinalReserve
AMTOK = CB <= MAXCB

'Note
```

```
'        Final Answer--> StrucOK & AmtOK
'*********************************************************************
***

Set BPMod = Nothing

If Err.Number <> 0 then
  SystemError = Err.Description
End if

'%>
```

10043579 .010502

```
'<%Template=California%> <%Version=04.30.2001%>


function dollarString(no, n)
  if (no >= 0.00) and (no <= 1.001) then
    dollarString = formatnumber(no, n) + " dollar "
  else
    dollarString = formatnumber(no, n) + " dollars "
  end if
end function

ErrDisp = ""

function LookupIns ( vAmt, vCol )
        if vAmt < 0 then
                LookupIns = 0
        else
                Select Case vCol
                Case 1
              select case true
              Case ( vAmt >= 0 and vAmt <= 500  ) X =       0
              Case ( vAmt >= 501 and vAmt <= 750  ) X =      501
              Case ( vAmt >= 751 and vAmt <= 1000  ) X =     751
              Case ( vAmt >= 1001 and vAmt <= 1200  ) X =    1001
              Case ( vAmt >= 1201 and vAmt <= 1400  ) X =    1201
              Case ( vAmt >= 1401 and vAmt <= 1600  ) X =    1401
              Case ( vAmt >= 1601 and vAmt <= 1800  ) X =    1601
              Case ( vAmt >= 1801 and vAmt <= 2000  ) X =    1801
              Case ( vAmt >= 2001 and vAmt <= 2200  ) X =    2001
              Case ( vAmt >= 2201 and vAmt <= 2400  ) X =    2201
              Case ( vAmt >= 2401 and vAmt <= 2600  ) X =    2401
              Case ( vAmt >= 2601 and vAmt <= 2800  ) X =    2601
              Case ( vAmt >= 2801 and vAmt <= 3000  ) X =    2801
              Case ( vAmt >= 3001 and vAmt <= 3200  ) X =    3001
              Case ( vAmt >= 3201 and vAmt <= 3400  ) X =    3201
              Case ( vAmt >= 3401 and vAmt <= 3600  ) X =    3401
              Case ( vAmt >= 3601 and vAmt <= 3800  ) X =    3601
              Case ( vAmt >= 3801 and vAmt <= 4000  ) X =    3801
              Case ( vAmt >= 4001 and vAmt <= 4200  ) X =    4001
              Case ( vAmt >= 4201 and vAmt <= 4400  ) X =    4201
              Case ( vAmt >= 4401 and vAmt <= 4600  ) X =    4401
              Case ( vAmt >= 4601 and vAmt <= 4800  ) X =    4601
              Case ( vAmt >= 4801 and vAmt <= 5000  ) X =    4801
              Case ( vAmt >= 5001 and vAmt <= 5200  ) X =    5001
              Case ( vAmt >= 5201 and vAmt <= 5400  ) X =    5201
              Case ( vAmt >= 5401 and vAmt <= 5600  ) X =    5401
              Case ( vAmt >= 5601 and vAmt <= 5800  ) X =    5601
              Case ( vAmt >= 5801 and vAmt <= 6000  ) X =    5801
              Case ( vAmt >= 6001 and vAmt <= 6500  ) X =    6001
              Case ( vAmt >= 6501 and vAmt <= 7000  ) X =    6501
              Case ( vAmt >= 7001 and vAmt <= 7500  ) X =    7001
              Case ( vAmt >= 7501 and vAmt <= 8000  ) X =    7501
              Case ( vAmt >= 8001 and vAmt <= 8500  ) X =    8001
              Case ( vAmt >= 8501 and vAmt <= 9000  ) X =    8501
              Case ( vAmt >= 9001 and vAmt <= 9500  ) X =    9001
              Case ( vAmt >= 9501 and vAmt <= 10000  ) X =   9501
              Case ( vAmt >= 10001 and vAmt <= 1000000 ) X =  10001
              End Select
                Case 2
              select case true
              Case ( vAmt >= 0 and vAmt < 501  ) X =        173
              Case ( vAmt >= 501 and vAmt < 751  ) X =      209
              Case ( vAmt >= 751 and vAmt < 1001  ) X =     250
        Case ( vAmt >= 1001 and vAmt < 1201  ) X =     278
        Case ( vAmt >= 1201 and vAmt < 1401  ) X =     304
        Case ( vAmt >= 1401 and vAmt < 1601  ) X =     327
        Case ( vAmt >= 1601 and vAmt < 1801  ) X =     344
        Case ( vAmt >= 1801 and vAmt < 2001  ) X =     366
        Case ( vAmt >= 2001 and vAmt < 2201  ) X =     390
        Case ( vAmt >= 2201 and vAmt < 2401  ) X =     409
        Case ( vAmt >= 2401 and vAmt < 2601  ) X =     431
        Case ( vAmt >= 2601 and vAmt < 2801  ) X =     446
        Case ( vAmt >= 2801 and vAmt < 3001  ) X =     467
        Case ( vAmt >= 3001 and vAmt < 3201  ) X =     486
        Case ( vAmt >= 3201 and vAmt < 3401  ) X =     509
        Case ( vAmt >= 3401 and vAmt < 3601  ) X =     526
        Case ( vAmt >= 3601 and vAmt < 3801  ) X =     545
        Case ( vAmt >= 3801 and vAmt < 4001  ) X =     564
        Case ( vAmt >= 4001 and vAmt < 4201  ) X =     586
        Case ( vAmt >= 4201 and vAmt < 4401  ) X =     599
        Case ( vAmt >= 4401 and vAmt < 4601  ) X =     620
        Case ( vAmt >= 4601 and vAmt < 4801  ) X =     635
        Case ( vAmt >= 4801 and vAmt < 5001  ) X =     655
        Case ( vAmt >= 5001 and vAmt < 5201  ) X =     672
        Case ( vAmt >= 5201 and vAmt < 5401  ) X =     686
        Case ( vAmt >= 5401 and vAmt < 5601  ) X =     703
        Case ( vAmt >= 5601 and vAmt < 5801  ) X =     721
        Case ( vAmt >= 5801 and vAmt < 6001  ) X =     738
        Case ( vAmt >= 6001 and vAmt < 6501  ) X =     783
        Case ( vAmt >= 6501 and vAmt < 7001  ) X =     826
        Case ( vAmt >= 7001 and vAmt < 7501  ) X =     873
        Case ( vAmt >= 7501 and vAmt < 8001  ) X =     919
        Case ( vAmt >= 8001 and vAmt < 8501  ) X =     967
        Case ( vAmt >= 8501 and vAmt < 9001  ) X =     1015
        Case ( vAmt >= 9001 and vAmt < 9501  ) X =     1060
        Case ( vAmt >= 9501 and vAmt < 10001  ) X =    1107
        Case ( vAmt >= 10001 and vAmt <= 1000000 ) X =  1107
        End Select
Case 3
        select case true
        Case ( vAmt >= 0 and vAmt < 501  ) X =         0.005605
        Case ( vAmt >= 501 and vAmt < 751  ) X =       0.005605
        Case ( vAmt >= 751 and vAmt < 1001  ) X =      0.005605
        Case ( vAmt >= 1001 and vAmt < 1201  ) X =     0.005605
        Case ( vAmt >= 1201 and vAmt < 1401  ) X =     0.00817
        Case ( vAmt >= 1401 and vAmt < 1601  ) X =     0.00817
        Case ( vAmt >= 1601 and vAmt < 1801  ) X =     0.00817
        Case ( vAmt >= 1801 and vAmt < 2001  ) X =     0.00817
        Case ( vAmt >= 2001 and vAmt < 2201  ) X =     0.00817
        Case ( vAmt >= 2201 and vAmt < 2401  ) X =     0.01083
        Case ( vAmt >= 2401 and vAmt < 2601  ) X =     0.01083
        Case ( vAmt >= 2601 and vAmt < 2801  ) X =     0.01083
        Case ( vAmt >= 2801 and vAmt < 3001  ) X =     0.01083
        Case ( vAmt >= 3001 and vAmt < 3201  ) X =     0.01083
        Case ( vAmt >= 3201 and vAmt < 3401  ) X =     0.01349
        Case ( vAmt >= 3401 and vAmt < 3601  ) X =     0.01349
        Case ( vAmt >= 3601 and vAmt < 3801  ) X =     0.01349
        Case ( vAmt >= 3801 and vAmt < 4001  ) X =     0.01349
        Case ( vAmt >= 4001 and vAmt < 4201  ) X =     0.01349
        Case ( vAmt >= 4201 and vAmt < 4401  ) X =     0.01349
        Case ( vAmt >= 4401 and vAmt < 4601  ) X =     0.015067
        Case ( vAmt >= 4601 and vAmt < 4801  ) X =     0.015067
        Case ( vAmt >= 4801 and vAmt < 5001  ) X =     0.015067
        Case ( vAmt >= 5001 and vAmt < 5201  ) X =     0.015067
        Case ( vAmt >= 5201 and vAmt < 5401  ) X =     0.015067
        Case ( vAmt >= 5401 and vAmt < 5601  ) X =     0.016777
```

```
                Case ( vAmt >= 5601 and vAmt < 5801  ) X =    0.016777
                Case ( vAmt >= 5801 and vAmt < 6001  ) X =    0.016777
                Case ( vAmt >= 6001 and vAmt < 6501  ) X =    0.016777
                Case ( vAmt >= 6501 and vAmt < 7001  ) X =    0.018297
                Case ( vAmt >= 7001 and vAmt < 7501  ) X =    0.018297
                Case ( vAmt >= 7501 and vAmt < 8001  ) X =    0.019627
                Case ( vAmt >= 8001 and vAmt < 8501  ) X =    0.019627
                Case ( vAmt >= 8501 and vAmt < 9001  ) X =    0.020767
                Case ( vAmt >= 9001 and vAmt < 9501  ) X =    0.020767
                Case ( vAmt >= 9501 and vAmt < 10001  ) X =   0.020767
                Case ( vAmt >= 10001 and vAmt <= 1000000 ) X =  0.020767
                End Select
                  Case Else ErrDisp = "Expression Error on INS lookup - Column
selected: " & vCol
                  End Select
      LookupIns = X
         end if
end function

function LookupJobTable ( vAmt, vCol )
      if vAmt < 0 then
            LookupJobTable = 0
      else
            Select Case vCol
            Case 1
         Select Case True
         Case ( vAmt >=  0 And vAmt < 0.5 )    X = 0
         Case ( vAmt >=  0.5 And vAmt < 1 )    X = 0.5
         Case ( vAmt >=  1 And vAmt < 2.5 )    X = 1
         Case ( vAmt >=  2.5 And vAmt < 4.5 )  X = 2.5
         Case (vAmt >= 4.5)                    X = 4.5
         end Select
            Case 2
         Select Case True
         Case ( vAmt >=  0 And vAmt < 0.5 )    X = 0.11
         Case ( vAmt >=  0.5 And vAmt < 1 )    X = 0.3515
         Case ( vAmt >=  1 And vAmt < 2.5 )    X = 0.685
         Case ( vAmt >=  2.5 And vAmt < 4.5 )  X = 2.68
         Case (vAmt >= 4.5)                    X = 10
         end Select
            Case 3
         Select Case True
         Case ( vAmt >=  0 And vAmt < 0.5 )    X = 0.483
         Case ( vAmt >=  0.5 And vAmt < 1 )    X = 0.667
         Case ( vAmt >=  1 And vAmt < 2.5 )    X = 1.33
         Case ( vAmt >=  2.5 And vAmt < 4.5 )  X = 3.66
         Case (vAmt >= 4.5)                    X = 0
         end Select
            Case 4
         Select Case True
         Case ( vAmt >=  0 And vAmt < 0.5 )    X = 0.5
         Case ( vAmt >=  0.5 And vAmt < 1 )    X = 0.5
         Case ( vAmt >=  1 And vAmt < 2.5 )    X = 1.5
         Case ( vAmt >=  2.5 And vAmt < 4.5 )  X = 2
         Case (vAmt >= 4.5)                    X = 999995.5
         end Select
            Case Else ErrDisp = "Expression Error on JobLookup - Column
selected: " & vCol
            End Select
      LookupJobTable = X
         end if
end function
```

```
function LookupCFScalerTable( vAmt, vCol )
        if vAmt < 0 then

                LookupCFScalerTable = 0
        else
                Select Case vCol
                Case 1
            Select Case True
            Case ( vAmt >=  0 And vAmt < 1 )      X = 0
            Case ( vAmt >=  1 And vAmt < 2 )      X = 1
            Case ( vAmt >=  2 And vAmt < 2.75 )   X = 2
            Case ( vAmt >=  2.75 And vAmt < 3.5 ) X = 2.75
            Case (vAmt >= 3.5)                     X = 3.5
            End Select
                Case 2
            Select Case True
            Case ( vAmt >=  0 And vAmt < 1 )      X = 0.5
            Case ( vAmt >=  1 And vAmt < 2 )      X = 0.7
            Case ( vAmt >=  2 And vAmt < 2.75 )   X = 0.96
            Case ( vAmt >=  2.75 And vAmt < 3.5 ) X = 1.08
            Case (vAmt >= 3.5)                     X = 1.28
            End Select
                Case 3
            Select Case True
            Case ( vAmt >=  0 And vAmt < 1 )      X = 0.2
            Case ( vAmt >=  1 And vAmt < 2 )      X = 0.18
            Case ( vAmt >=  2 And vAmt < 2.75 )   X = 0.16
            Case ( vAmt >=  2.75 And vAmt < 3.5 ) X = 0.32
            Case (vAmt >= 3.5)                     X = 0.16
            End Select
                Case Else ErrDisp = "Expression Error on CFS lookup - Column
selected: " & vCol
                End Select
    LookupCFScalerTable = X
        end if
end function

function LookupTermTable( vClass, vCol )
        if vClass < 0 then
                LookupTermTable = 0
        else
                Select Case vCol
                Case 1
            Select Case vClass
            Case 0 X = 0
            Case 1 X = 1
            Case 2 X = 2
            Case 3 X = 3
            Case 4 X = 4
            Case 5 X = 5
            End Select
                Case 2
            Select Case vClass
            Case 0 X = 150000
            Case 1 X = 150000
            Case 2 X = 140000
            Case 3 X = 140000
            Case 4 X = 110000
            Case 5 X = 110000
            End Select
                Case 3
            Select Case vClass
        Case 0 X = 7
        Case 1 X = 7
        Case 2 X = 6
        Case 3 X = 6
        Case 4 X = 5
        Case 5 X = 5
        End Select
            Case 4
        Select Case vClass
        Case 0 X = 6
        Case 1 X = 6
        Case 2 X = 5
        Case 3 X = 5
        Case 4 X = 3
        Case 5 X = 3
        End Select
Case 5
        Select Case vClass
        Case 0 X = 110000
        Case 1 X = 110000
        Case 2 X = 100000
        Case 3 X = 100000
        Case 4 X = 70000
        Case 5 X = 70000
        End Select
            Case 6
        Select Case vClass
        Case 0 X = 9
        Case 1 X = 9
        Case 2 X = 7
        Case 3 X = 7
        Case 4 X = 5
        Case 5 X = 5
        End Select
            Case 7
        Select Case vClass
        Case 0 X = 150000
        Case 1 X = 150000
        Case 2 X = 130000
        Case 3 X = 130000
        Case 4 X = 110000
        Case 5 X = 110000
        End Select
            Case 8
        Select Case vClass
        Case 0 X = 1.4
        Case 1 X = 1.3
        Case 2 X = 1.2
        Case 3 X = 1.1
        Case 4 X = 1
        Case 5 X = 0.9
        End Select
            Case 9
        Select Case vClass
        Case 0 X = 1800
        Case 1 X = 1750
        Case 2 X = 1500
        Case 3 X = 1100
        Case 4 X = 0
        Case 5 X = 0
        End Select
            Case 10
        Select Case vClass
```

20050104562

```
            Case 0 X = 0.1
            Case 1 X = 0.1
            Case 2 X = 0.1
            Case 3 X = 0.05
            Case 4 X = 0.05
            Case 5 X = 0.05
            End Select
                Case Else ErrDisp = "Expression Error on TERM lookup - Column
selected: " & vCol
                End Select
    LookupTermTable = X
        end if
end function

function LookupApr( vTerm )
    if vTerm < 1  then ErrDisp = "TERM cannot be less than 1 - Current TERM is: " &
vTerm
    if vTerm > 48 then ErrDisp = "TERM cannot be greater than 48 - Current TERM is:
" & vTerm
        Select Case vTerm
  Case 1    LookupApr = 0.12
  Case 2    LookupApr = 0.1596
  Case 3    LookupApr = 0.1791
  Case 4    LookupApr = 0.1905
  Case 5    LookupApr = 0.1978
  Case 6    LookupApr = 0.2029
  Case 7    LookupApr = 0.2064
  Case 8    LookupApr = 0.2091
  Case 9    LookupApr = 0.2111
  Case 10   LookupApr = 0.2126
  Case 11   LookupApr = 0.2137
  Case 12   LookupApr = 0.2146
  Case 13   LookupApr = 0.2152
  Case 14   LookupApr = 0.2157
  Case 15   LookupApr = 0.216
  Case 16   LookupApr = 0.2162
  Case 17   LookupApr = 0.2164
  Case 18   LookupApr = 0.2164
  Case 19   LookupApr = 0.2164
  Case 20   LookupApr = 0.2164
  Case 21   LookupApr = 0.2163
  Case 22   LookupApr = 0.2161
  Case 23   LookupApr = 0.2159
  Case 24   LookupApr = 0.2157
  Case 25   LookupApr = 0.2155
  Case 26   LookupApr = 0.2152
  Case 27   LookupApr = 0.2115
  Case 28   LookupApr = 0.2146
  Case 29   LookupApr = 0.2144
  Case 30   LookupApr = 0.2141
  Case 31   LookupApr = 0.2137
  Case 32   LookupApr = 0.2134
  Case 33   LookupApr = 0.2131
  Case 34   LookupApr = 0.2127
  Case 35   LookupApr = 0.2124
  Case 36   LookupApr = 0.212
  Case 37   LookupApr = 0.2117
  Case 38   LookupApr = 0.2113
  Case 39   LookupApr = 0.2109
  Case 40   LookupApr = 0.2105
  Case 41   LookupApr = 0.2102
  Case 42   LookupApr = 0.2098
  Case 43   LookupApr = 0.2094
  Case 44   LookupApr = 0.209
  Case 45   LookupApr = 0.2087
  Case 46   LookupApr = 0.2083
  Case 47   LookupApr = 0.20779
  Case 48   LookupApr = 0.2075
  Case Else ErrDisp = "Error on APR lookup - TERM is: " & vTerm
  End Select
end function


'*****************************************************************
' function ErrLookup
'*****************************************************************
function ErrLookup( vErr)
        if vErr < 0  then ErrDisp = "Error on ERR lookup - ERR is: " & vErr
        if vErr >= 99 then
                ErrLookup = ""
        else
        Select Case vErr
  Case 0 ErrLookup = "DEAL STRUCTURE IS UNACCEPTABLE"
                hint = "Deal structure is unacceptable.  " + hint1 + "  You will
need more down and a lower price.  Or maybe a really inexpensive car.  " + hint11 +
hint17 + hint18 + hint20
  Case 1 ErrLookup = "MAXIMUM TERM 48 MONTHS"
                hint = "YOU CAN'T GO LONGER THAN 48 MONTHS."
  Case 2 ErrLookup = "NEED SHORTER TERM!"
                hint = "Need Shorter Term!"
  Case 3 ErrLookup = "PAYMENT MUST BE HIGHER THAN $140!"
                hint= "The payment is too low.  This is a car, not a couch."
  Case 4 ErrLookup = "TRY 28 MONTH OR SHORTER TERM"
  Case 5 ErrLookup = "MINIMUM DISCOUNT 10% OF AMOUNT FINANCED LESS INS"
                hint= "The discount has to be at least 10 percent."
  Case 6 ErrLookup = "MINIMUM DISCOUNT $300"
                hint = "Minimum discount is 300 dollars"
  Case 7 ErrLookup = "AMOUNT FINANCED IN EXCESS OF MAX ALLOWED BY:"
                hint =  ""
+hint25+hint21+hint23+hint24+hint8+hint20+hint1+hint2+hint3+ hint4 + hint5 +hint6+
hint7 + hint9 + hint10 + hint12 + hint13 + hint14 + hint15 + hint16 + hint17 +
hint18
  Case 8 ErrLookup = "SUGGEST 28 MONTH MAX TERM"
                hint= "You can make a better deal if you lower the term to 28
months."
  Case 9 ErrLookup = "COX IS MAKING CUSTOMER FACTOR LOWER"
                hint = "Co-signer is making the customer factor lower.  Please check
to see if the co signer is actually the buyer."
  Case 10 ErrLookup = ""
                hint = hint8+hint20+hint1+hint2+hint3+ hint4 + hint5 +hint6+ hint7 +
hint9 + hint10 + hint12 + hint13 + hint14 + hint15 + hint16 + hint17 + hint18
  Case 11 ErrLookup = "AMOUNT FINANCED IN EXCESS OF MAX ALLOWED BY:"
                hint = "" + hint22
  Case 12 ErrLookup = "AMOUNT FINANCED IN EXCESS OF MAX ALLOWED BY:"
                hint = "You need to put down  " + dollarString(OA,0) + " more."

  End Select
        end if
End Function
```

VEHICLE CLASSIFICATION SHEET 11/2000

IMPORTS

ACURA

Integra Man Trans	3
Legend 86-90	5
Vigor	3
All Others	1

DAEWOO

4 Dr + Auto	3
All Others	4

HONDA

Civic 92-newer 4dr+Auto.	"S"
Civic 92-newer 4dr+Man.	1
Other Civic Automatic	2
Other Civic Man Trans	3
CRX/Prelude	3
Accord 89&Older	3
Accrd 91-94 LX 4dr+Auto	"S"
All Others inc Accrd Wgn.	1

HYUNDAI

Scoupe (All)	5
Other 97+newer	4
All Others	3

ISUZU

Pickups	1
Trooper/Rodeo 4dr+Auto.	2
Trooper/Rodeo Other	3
All Others	5

LEXUS

All	3

MAZDA CARS

MX-6	5
Miata	5
Protégé 94-older	4
RX7	5
929 91-older	4
All Others	3

MAZDA TRUCKS

Pickups Auto+Xcab	2
Navajo	4
All Others	3

MITSUBISHI

Galant 94 & newer	3
Montero	3
Pickups	1
All Others	5

NISSAN CARS

Altima 93-95 w Auto	2
Maxima 89&newer Auto	2
Sentra 92-older	"S"
Sentra 93-newer	1
240SX	4
300 ZX	5
All Others	3

NISSAN TRUCKS

Pathfinder 4 Dr + Auto	1
Pickups	"S"
Quest	2
All Others	3

VEHICLES 10 YRS OLD OR MORE: Add 100,000 miles to odometer if a 5 digit odometer. 6 digit odometer vehicles must be booked with at least 100,000 miles.

IMPORTS (Cont)

TOYOTA CARS

Camry 92-93 Auto	"S"
Celica/Cressida/MR2	3
Corolla 93-94 Auto	"S"
Supra	5
All Others	1

TOYOTA TRUCKS

Pickups	"S"
4-Runner 90-91 V6+4dr+Auto	"S"
Vans 89 & older	4
All Others	1

VOLKSWAGEN

Jetta/Passat 4 Dr	3
All Others	5

DOMESTICS

BUICK

Quad 4, Tech 4 or	5
Regal 92&newer w 3.8L	2
Other 92&newer w 3.8L	3
Century/Skylark/Regal	3
All Others	4

CHEVROLET

Quad 4, Tech 4 or 2.8L	5
Camaro	5
Corvette	5
Corsica/Caprice	4
All Others	3

CHEVROLET / GMC TRUCKS

Astro/Safari 2WD	1
Blazer 4dr+4.3L 95+	2
S10 Blazer 2dr All	5
C-Series w Auto	1
C-Series Other	2
K-5 Blazer/Tahoe/Yukon	1
Lumina Van	5
S10 X-Cab 4.3L+Auto	1
Suburban	2
All Others	3

CHRYSLER

Cirrus	3
Concorde	4
Town & Country	5
All Others	5

DODGE/PLYMOUTH CARS

Turbos/Convertibles	5
Intrepid	4
Neon 4 dr + Auto	3
Shadow/Sundance	3
Spirit/Acclaim	3
Stratus/Breeze	3
All Others	5

DODGE / PLYMOUTH TRUCKS

Caravan/Voyager 96-newer 2WD	3
Caravan/Voygr Other	5
94+ Trucks V-8	2
Dakota V6/V8	2
All Others	3

DOMESTICS

FORD CARS

Turbo/Supercharger	5
Escort	4
Mustang 94 & newer	2
Taurus Sedan 95 & older	5
Taurus Wagon	5
T-Bird 90-93	2
All Others	3

FORD TRUCKS

Aerostar 4X4	5
Explorer 4 Dr + Auto	2
Explorer Other	4
F Series Auto + V-8	1
F Series Other	2
Ranger X Cab 6 cyl + Auto	1
Ranger 6 cyl + Auto	2
All Others	3

GEO

Prism 4dr Sedan w Auto	1
Prism 4dr Sedan w Man	2
Tracker	5
All Others	3

JEEP

CJ & Wrangler 6 cyl	1
Other CJ/Wrangler	3
Cherokee 4dr+4.0L+Auto	3
Grand Cherokee	3
All Others	5

LINCOLN

Towncar	4
All Others	5

MERCURY

Capri	5
Tracer	4
Sable Wagon	5
Sable Sedan 95 & older	5
All Others	3

OLDSMOBILE

Quad 4, Tech 4	5
Silhouette	5
All Other 3.8L or V8	4
All Other 4 or 6 cyl	3

PONTIAC

Quad 4, Tech 4	5
Firebird	5
Transport	5
All Others	3

SATURN

All	3

EXAMPLE OF CONTENTS OF A CLASS TABLE

ADDITIONAL POLICIES

1. ANY VEHICLE NOT LISTED SHALL BE CONSIDERED CLASS 5.

2. DO NOT ADD FOR LOW MILES, OR "SOFT ADDS."

WESTLAKE WILL NOT ADVANCE FOR THE FOLLOWING KELLEY ADDS: PREMIUM SOUND, PREMIUM WHEELS, ABS, DUAL AIR BAGS, INTEGRATED PHONE, UPGRADED TOPS, BUMPER, OR PAINT, WIDE/OVERSIZE TIRES, TOW PACKAGE, GRILLE GUARD, WINCH, COMMERCIAL TRUCK ADDS & ANY ITEM NOT IN WORKING ORDER.

3. ANY VARIANCE FOUND BETWEEN ACTUAL & REPRESENTED VALUE OF THE VEHICLE MAY RESULT IN DEALER REPURCHASE.

```
'<%Template=California%> <%Version=11.01.2001%>
' California Expression Template
' Modification Date : Nov 16, 2000
' Reason: converted from Delphi to VB Script
' Modification Date : Nov 17, 2000
' Reason: Added code for COM, modified for Stand Alone BP
' Modification Date : Nov 22, 2000
' Reason: Added TotalofPayments calculation
' Modification Date : Jan 25, 2001
' Reason: Added insuarnce cap beyond $10,000.00
' Modification Date : Feb 13, 2001
' Reason: Repaired wizard re o/a, etc
' Modification Date : Feb 26, 2001 - John Sun
' Reason: Added error handling - when error occurs, system need to continue and trap

' all the error messages.
' Modification Date : Mar 26, 2001 - Mike Duke
' Reason: Repaired Ins Lookup Table to account for all Carryback possibilities.
' Modification Date : Apr 03, 2001
' Reason: Made minimum Total Income = $1.00
' Modification Date : Apr 09, 2001
' Reason: Move Big Mile Hit expressions in proper order for proper recalc when
opening saved deal
' Modification Date : Apr 30, 2001
' Reason: Fix error in Job Lookup Table
' Modification Date : May 16, 2001
' Reason: Fix error in CF Scaler Lookup
' Modification Date : May 23, 2001
' Reason: Allow Class 5 for reserve deals
' Modification Date : Sept 10, 2001
' Reason: Re-sequence MinBk Module
' Modification Date : Nov 1, 2001
' Reason: Complete resorting of expressions
'--------------------------------Added for Stand Alone
BP--------------------------------


On Error Resume Next
Set BPMod = CreateObject("BPfunctionsModule.BPFunctions")
'-----------------------------------------------------------------------------------
-----

'[CONSTANTS]

'System Error
DIM SystemError
SystemError = ""
Acqfee=100
TradeScaler=0.70

HCBAmtFin = 8000
HCBScaler = - 0.00015

StupidNum = 8
StupidTerm = 17

FTBINC = 1
FTBPMTRatio = 1
FTBCB = 1
FTBPhBill = 2
FTBDown = 0.20
FTBSpouse = 2
FTBResid = 3
```

```
FTBJob = 3

STHit1 = 0.9
STHit2 = 0.6
STScaler1 = - 0.10
STScaler2 = - 0.20

SpreadNumScaler = 0.50

OptimalCB = 5800
AllowVariance = 1700
OptimalPoints = 0.13

CostPerKinkPoint = 2
KinkAge = 8
KinkMiles = 120000
KinkCF = 1.70
KinkMaxTerm = 28

CFTRWScaler = 0.75
CFJobScaler = 0.90
CFResidScaler = 0.60
CFHiGoodScaler = 0.90
CFBKScaler = 1.00
CFHomeScaler = 0.80
CFIncScaler = 0.075
CFDebtScaler = 1.00

FreeGetNone = 1.75
FreeGetAll = 2.30
SBGetNone = 2.75
SBGetAll = 3.25

BMHiLimit = 6000
BMLowLimit = 2000

MCBHiMiles = 140000
MCBHiMilesRange = 10000
MCBMaxIns = 1000

SDDollarDown = 1500
SDPercentDown = 0.30
SDScaler = 0.80
SDEquityMult = 0.50

BKStrong = 0.5
BKGood = 0.2
BKInc = 0.2
BKSpouse = 0.05
BKHiGood = 0.2
CRStart = 0.15
CRCountAll = 0.20

DATerm = 30
DAScaler = 0.90
MinRent = 250
SpreadReq = 0.12
TooSmallPmt = 140

MBKDown = 1
MBKInc = 3
MBKHome = 1
MBKSpouse = 1
```

```
MBKMinPoints = 6
MinDiscStrongBK = 0.11
MinDiscRegularBK = 0.15
MBKGood = 3
MBKHiGood = 3

BSSmallHD = - 0.20
BSBK = - 0.20
BadScalerBase = 1.05
BSHiDerog = 0.20
MinDiscHiDerog = 0.12
MinDiscount = 0.10

GSJustForPlaying = 1.50
GSHiGood = 0.25
GS2ManyAcct = - 0.25
GSGood2xDerog = 0.25
GSDerog2xGood = - 0.25
GSDerog5xGood = - 0.25
GSNoDerog = 0.20
GSFTB = 0.65
GSGoodMoreThanDerog = 0.10

BigMilesStart = 185000
BigMilesRange_1 = 50000
BigMilesRange_2 = 50000
BigMilesRange_3 = 50000
HitBigMiles_1 = 0.15
HitBigMiles_2 = 0.15
HitBigMiles_3 = 0.15

MaxWarrCB = 250
CurrYear = 2000

'[FIX YEAR OF CAR IN CASE USER INPUTS 2 DIGIT MODEL YEAR]
if (vYear < 5) then
  CarYear = vYear + 2000
  else if (vYear < 100) then
    CarYear = vYear + 1900
    else
      CarYear = vYear
  end if
end if

'[INITIALIZE HINTS]
hint =""
hint1=""
hint2=""
hint3=""
hint4=""
hint5=""
hint6=""
hint7=""
hint8=""
hint9=""
hint10=""
hint11=""
hint12=""
hint13=""
hint14=""
hint15=""
hint16=""
hint17=""
```

```
hint18=""
hint19=""
hint20=""
hint21=""
hint22=""
hint23=""
hint24=""
hint25=""


'[DEAL STRUCTURE CALCULATION AREA]
'[CALCULATE TAX AMOUNT AND SUBTOTAL]
Tax = (TaxRate/100) * (Price + Smog + Doc)
SubTot = cdbl(Price + Doc + Smog + SmogCert + Tax + LicFee + Warr)
TotalDown = Down + TradeAllowance - TradePayoff
TotalLessIns = SubTot - TotalDown

'[CALCULATE INSURANCE AMOUNT IF NEEDED]
if (InsFlag = 1) then
  if (TotalLessIns <= 10000)  then
    Ins = LookupIns(TotalLessIns, 2 )
  else
    Ins = 0.1088*TotalLessIns+95
  end if
else
  Ins = 0.00
end if

'[THIS IS THE AMOUNT FINANCED]
CB = (SubTot - TotalDown) + Ins

'[LOOKUP INTEREST RATE]
Interest = LookupApr( Term )
APR = Interest

'[CALCULATE PAYMENT]
PaymentA = BPMod.bp_AddOnPMT( CB, Term, 0.12, DaysToPay )
Payment = BPMod.bp_Trunc( PaymentA,2 )

'["ADDON" IS THE TOTAL DOLLAR AMOUNT OF INTEREST]
IntCost = ( Payment * Term ) - CB
AddOn = (Payment * Term) - CB
TotalofPayments= Payment*Term
FrGross = Price - Cost
DealerGross=PRICE-COST-RESERVE+WARR-WARCOST-AcqFee
'[END DEAL STRUCTURE CALCULATION AREA]



'[MAX AMOUNT FINANCED CALCULATION AREA = "MAXCB"]
'[CALCULATE HIT FOR VERY HIGH MILES = "BIGMILEHIT"]
LotsOfMiles_1 = BigMilesStart - ( vClass * 10000 )
LotsOfMiles_2 = LotsOfMiles_1 + BigMilesRange_1
LotsofMiles_3 = LotsOfMiles_2 + BigMilesRange_2

HitRate_1 = ( HitBigMiles_1 + ( vClass / 100 ) ) / BigMilesRange_1
HitRate_2 = ( HitBigMiles_2 + ( vClass / 100 ) ) / BigMilesRange_2
HitRate_3 = ( HitBigMiles_3 + ( vClass / 100 ) ) / BigMilesRange_3

BigMileDelta_2 = BPMod.bp_MIN( Miles - LotsOfMiles_2, BigMilesRange_2 ) * HitRate_2
BigMileDelta_3 = BPMod.bp_MIN( Miles - LotsOfMiles_3, BigMilesRange_3 ) * HitRate_3
BigMileHit_1 = BPMod.bp_MIN( Miles - LotsOfMiles_1, BigMilesRange_1 ) * HitRate_1
BigMileHit_2 = BPMod.bp_IFG( Miles, LotsOfMiles_2, BigMileHit_1 + BigMileDelta_2,
BigMileHit_1 )
BigMileHit_3 = BPMod.bp_IFG( Miles, LotsOfMiles_3, BigMileHit_2 + BigMileDelta_3,
BigMileHit_2 )
BigMileHit = BPMod.bp_IFG( Miles, LotsOfMiles_1, BigMileHit_3, 0 )

'[CALCULATE REGULAR HI MILE HIT = "HIMILEHIT"]
OverMiles = MCBHiMiles - MCBHiMilesRange
MaxHiMileHit = LookupTermTable( vClass, 10 )
MCBHitRate = MaxHiMileHit / MCBHiMilesRange
HiMileHitExp1 = BPMod.bp_MIN( ( Miles - OverMiles ), MCBHiMilesRange ) * MCBHitRate
* Book
HiMileHit = BPMod.bp_IFG( Miles,MCBHiMiles - MCBHiMilesRange,HiMileHitExp1, 0 )

'[GET MAXCB]
BMRange = BMHiLimit - BMLowLimit
CarClassAdv = LookupTermTable( vClass, 8 ) * Book
MaxBookAdv = LookupTermTable( vClass, 9 ) + Book
WarrAllowance = BPMod.bp_MIN( MaxWarrCB, Warr )
PossibleAdv = CarClassAdv - HiMileHit + WarrAllowance + BPMod.bp_MIN ( Ins,
MCBMaxIns )
OKAdv = BPMod.bp_MIN( PossibleAdv, ( MaxBookAdv + Ins + WarrAllowance ) )
BigMileSmackScaler = BPMod.bp_MAX( BPMod.bp_MIN( ( OKAdv - Ins - BMLowLimit ) /
BMRange, 1 ),0 )
BigMileSmack = ( OKAdv - Ins ) * BigMileHit * BigMileSmackScaler
MaxAltCB = 1500 + Ins - 100 * ( CurrYear - BPMod.bp_MIN( CurrYear, CarYear ) - 10 )
MaxCB = BPMod.bp_MAX( ( OKAdv - BigMileSmack ), MaxAltCB )
'[END MAX AMOUNT FINANCED CALCULATION AREA]


'[ASSORTED ONE-LINE VARIABLE CALCULATIONS FOR FUTURE USE]
RealDown = Down + (TradeAllowance - TradePayoff)*TradeScaler
CarAge = CurrYear - CarYear
EquityTest = TotalLessIns / ( MaxCB - Ins )


'[CALCULATE GOOD/DEROG INCLUDING SPOUSE]
TotalGood = BPMod.bp_IFB( Spouse,( Good + SpGood ) / 2, Good )
TotalDerog = BPMod.bp_IFB( Spouse, ( Derog + SpDerog ) / 2, Derog )
RealHiGood = BPMod.bp_IFB( Spouse, BPMod.bp_MAX( HiGood, SpHiGood ), HiGood )
RealHiDerog = BPMod.bp_IFB( Spouse, BPMod.bp_MAX( HiDerog, SpHiDerog ), HiDerog )

'[CALCULATE INCOME INCLUDING SPOUSE = "REALINC"]
TotalInc=BPMod.bp_MAX(BPMod.bp_IFB(Spouse, Inc+SpInc-Support, Inc-Support), 1)
RealIncCond1 = BPMod.bp_IFG( TotalGood, 1.5, 1, 0 )
RealIncCond2 = BPMod.bp_IFL( TotalDerog, TotalGood, 1,0 )
RealIncCond3 = BPMod.bp_IFGE( YrsTRW, 2, 1,0 )
RealIncCond4 = BPMod.bp_IFLE( TotalDerog, 2, 1, 0 )
RealIncCond5 = BPMod.bp_IFE( Repos, 0, 1,0 )
RealIncCond = RealIncCond1 * RealIncCond2 * RealIncCond3 * RealIncCond4 *
RealIncCond5
MinInc = BPMod.bp_MAX( Inc-Support, SpInc-Support )
IncHit = BPMod.bp_MAX( 1 - ( TotalInc / 10000 ), 0.75 )
RealIncExp2 = BPMod.bp_MAX( BPMod.bp_MAX( TotalInc * IncHit, TotalInc - 500 ),
MinInc )
RealIncExp1 = BPMod.bp_IFB( RealIncCond, TotalInc, RealIncExp2 )
RealIncExp = BPMod.bp_IFB( Spouse, RealIncExp1, TotalInc )
RealInc = BPMod.bp_MAX(RealIncExp, 1)


'[CALCULATE COXSCALER TO BE USED IF COX=YES]
GoodCreditExp = BPMod.bp_IFL( CoxGood,Good, - 2,0 )
GoodCreditPoints = BPMod.bp_IFB( BPMod.bp_IFG( CoxGood,Good,1,0 ) * BPMod.bp_IFGE(
CoxGood,4,1,0 ), 2, GoodCreditExp )
DerogExp = BPMod.bp_IFB( BPMod.bp_IFG( CoxDerog,3,1,0 ) + BPMod.bp_IFG(
CoxDerog,CoxGood,1,0 ), - 1, 0 )
DerogCreditPoints = BPMod.bp_IFB( BPMod.bp_IFLE( CoxDerog,CoxGood * 0.5, 1,0 ) *
BPMod.bp_IFLE( CoxDerog,3,1,0 ) * BPMod.bp_IFGE( CoxGood,1,1,0 ), 2, DerogExp )
RepoPoints = BPMod.bp_IFE( CoxRepo,0,1, - 10 * CoxRepo )
IncAccounts = BPMod.bp_MAX( (CoxGood + CoxDerog), 1 )
IncDivAcct = BPMod.bp_IFE( ( CoxGood + CoxDerog ), 0, 0, (CoxInc / IncAccounts) )
IncomePointsElseExp = BPMod.bp_IFB( BPMod.bp_IFGE( IncDivAcct,200,1,0 ) +
BPMod.bp_IFGE( CoxInc,4000,1,0 ), 3, ( ( IncDivAcct - 100 ) / 100 ) * 3 )
IncomePoints = BPMod.bp_IFLE( IncDivAcct,100,0, IncomePointsElseExp )
CoxOwnHomePoints = BPMod.bp_IFB( CoxHome,3,0 )
CoxParentOfBuyerPoint = BPMod.bp_IFB( CoxParent,5, - 1 )
BuyerLowOnBureauPointElseExp2 = BPMod.bp_IFLE( YrsTRW,3,0, - 1 )
BuyerLowOnBureauPointElseExp = BPMod.bp_IFLE( YrsTRW,2,1,
BuyerLowOnBureauPointElseExp2 )
BuyerLowOnBureauPoint = BPMod.bp_IFLE( YrsTRW,1,3,BuyerLowOnBureauPointElseExp )
CoxPoints = GoodCreditPoints + DerogCreditPoints + RepoPoints + IncomePoints +
CoxOwnHomePoints + CoxParentOfBuyerPoint + BuyerLowOnBureauPoint
GoodCoxExp1 = BPMod.bp_IFOR2( BPMod.bp_IFAND2( BPMod.bp_IFGE( CoxInc,1500,1,0 ),
BPMod.bp_IFGE( IncDivAcct,300,1,0 ), 1, 0 ), BPMod.bp_IFGE( CoxInc,2000,1,0 ), 1, 0
)
GoodCoxExp2 = BPMod.bp_IFE( CoxRepo, 0, 1,0 )
GoodCoxExp3 = BPMod.bp_IFL( CoxDerog, 3, 1, 0 )
GoodCoxExp4 = BPMod.bp_IFOR2( BPMod.bp_IFAND2( BPMod.bp_IFGE( CoxGood,5,1,0 ),
BPMod.bp_IFGE( CoxGood,5 * CoxDerog, 1,0 ),1,0 ),BPMod.bp_IFAND2( BPMod.bp_IFB(
CoxHome,1,0 ), BPMod.bp_IFLE( CoxDerog, 1, 1, 0 ),1,0 ), 1,0 )
GoodCoxCond = GoodCoxExp1 * GoodCoxExp2 * GoodCoxExp3 * GoodCoxExp4
GoodCoxInc = BPMod.bp_IFB( GoodCoxCond, BPMod.bp_MIN( ( CoxInc - 1500 ) / 1000, 1
),0 )
DerogNotZero = BPMod.bp_MAX( CoxDerog, 1 )
GoodCoxCredit = BPMod.bp_IFB( GoodCoxCond, BPMod.bp_IFB( CoxDerog, ( CoxGood /
DerogNotZero ) * 0.2, CoxGood * 0.2 ), 0 )
GoodCoxScaler = BPMod.bp_IFB( GoodCoxCond, BPMod.bp_MAX( GoodCoxCredit * GoodCoxInc,
1 ) * GoodCoxInc, 0 )
BadBuyer = BPMod.bp_IFB( BPMod.bp_IFG( YrsTRW + Derog,10,1,0 ) * BPMod.bp_IFG(
CoxPoints,0,1,0 ), 1, 0 )
BadBuyerScaler = BPMod.bp_IFB( BadBuyer, BPMod.bp_MAX( 0,1 - 0.1 * ( YrsTrw + Derog
- 10 ) ), 1 )
CoxScaler = BadBuyerScaler * ( CoxPoints + GoodCoxScaler * CoxPoints )


'[CALCULATE VARIABLE "RESIDTOT" FOR CUST FACT CALC LATER]
Resid8YearBase = BPMod.bp_IFGE( Resid, 8.1, BPMod.bp_MIN( Resid - 8, 4 ) * 0.00, 0 )
Resid5YearBase = BPMod.bp_IFGE( Resid, 5.1, BPMod.bp_MIN( Resid - 5, 3 ) * 1.44, 0 )
Resid1YearBase = BPMod.bp_IFGE( Resid, 1.1, BPMod.bp_MIN( Resid - 1, 4 ) * 1.27, 0 )
Resid0YearBase = BPMod.bp_IFGE( Resid, 0.0, BPMod.bp_MIN( Resid, 1 ) * 0.776, 0 )
ResidTot = Resid8YearBase + Resid5YearBase + Resid1YearBase + Resid0YearBase - 0.176


'[CALCULATE SCALER FOR GOOD/DEROG CREDIT ITEMS = "GOODSCALER", "BADSCALER"]
GoodScalerBase = GSJustForPlaying
GoodScaler9 = BPMod.bp_IFE( TotalDerog,0, GoodScalerBase + GSNoDerog, GoodScalerBase
)
GoodScaler8 = BPMod.bp_IFG( TotalGood,TotalDerog, GoodScaler9 + GSGoodMoreThanDerog,
GoodScaler9 )
GoodScaler7 = BPMod.bp_IFB( BPMod.bp_IFG( RealHiGood,RealHiDerog * 10,1,0 ) *
BPMod.bp_IFG( RealHiDerog,100,1,0 ) * BPMod.bp_IFL( RealHiDerog,3000,1,0
),GoodScaler8 + GSHiGood,GoodScaler8 )
GoodScaler6 = BPMod.bp_IFB( BPMod.bp_IFG( TotalGood,TotalDerog * 2,1,0 ) *
BPMod.bp_IFGE( TotalDerog,1,1,0 ),GoodScaler7 + GSGood2xDerog, GoodScaler7 )
GoodScaler5 = BPMod.bp_IFGE( TotalDerog,TotalGood * 2,GoodScaler6 +
GSDerog2xGood,GoodScaler6 )
```

```
GoodScaler4 = BPMod.bp_IFGE( TotalDerog,TotalGood * 5,GoodScaler5 +
GSDerog5xGood,GoodScaler5 )
GoodScaler3 = BPMod.bp_IFB( BPMod.bp_IFE( YrsTRW,0,1,0 ) * BPMod.bp_IFNE(
vClass,5,1,0 ),GoodScaler4 + GSFTB,GoodScaler4 )
GoodScaler2 = BPMod.bp_IFB( BPMod.bp_IFLE( YrsTrw,2,1,0 ) * BPMod.bp_IFGE( TotalGood
+ TotalDerog,6,1,0 ),GoodScaler3 + GS2ManyAcct,GoodScaler3 )
GoodScaler1 = BPMod.bp_IFL( RealHiDerog,1000,GoodScaler2 + ( 1000 - RealHiDerog ) *
0.0005, GoodScaler2 )
GoodScaler0 = BPMod.bp_IFL( YrsTRW,1,GoodScaler1 + ( 1 - YrsTRW ) * TotalGood * -
0.5, GoodScaler1 )
GoodScalerX = BPMod.bp_MIN ( GoodScaler0, 1.5 )
GoodScaler = BPMod.bp_MAX ( GoodScalerX, 0.25 )

BadScaler5 = BPMod.bp_IFB( BPMod.bp_IFGE( RealHiDerog,5000,1,0 ) * BPMod.bp_IFE(
BK,0,1,0 ), BadScalerBase + BSHiDerog, BadScalerBase )
BadScaler4 = BPMod.bp_IFB( BK, BadScaler5 + BSBK, BadScaler5 )
BadScaler3 = BPMod.bp_IFLE( RealHiDerog, 500, BadScaler4 + BSSmallHD, BadScaler4 )
BadScaler2 = BPMod.bp_IFB( BPMod.bp_IFB( BK,1,0 ) * BPMod.bp_IFL( YrsTRW,5,1,0 ),
BadScaler3 + (5 - YrsTRW)*0.3, BadScaler3 )
BadScaler1 = BPMod.bp_MAX( BadScaler2, 1.00 )
BadScaler = BPMod.bp_MIN( BadScaler1, 1.5 )

'[CALCULATE MIN % DISCOUNT BASED BK=YES AND OTHER FACTORS = "MINBK" ALSO TO BE USED
IN MINIMUM % DISCOUNT AREA BELOW]
BKPoints6 = BPMod.bp_IFGE( TotalDown,3000,MBKDown,0 )
BKPoints5 = BPMod.bp_IFGE( RealInc,3000,BKPoints6 + MBKInc,BKPoints6 )
BKPoints4 = BPMod.bp_IFGE( TotalGood,8,BKPoints5 + MBKGood,BKPoints5 )
BKPoints3 = BPMod.bp_IFB( Home,BKPoints4 + MBKHome, BKPoints4 )
BKPoints2 = BPMod.bp_IFB( Spouse,BKPoints3 + MBKSpouse,BKPoints3 )
BKPoints1 = BPMod.bp_IFGE( RealHiGood,10000, BKPoints2 + MBKHiGood,BKPoints2 )
MinBKExp2 = BPMod.bp_IFGE( BKPoints1, MBKMinPoints, MinDiscStrongBK,
MinDiscRegularBK )
MinBKCon = BPMod.bp_IFB( BK,1,0 ) * BPMod.bp_IFGE( RealInc,2400,1,0 ) *
BPMod.bp_IFGE( TotalGood,5,1,0 ) * BPMod.bp_IFGE( YrsTRW,8,1,0 ) * BPMod.bp_IFNE(
vClass,5,1,0 )
MinBKExp1 = BPMod.bp_IFB( MinBKCon, MinBKExp2,MinDiscRegularBK )
MinBK = BPMod.bp_IFB( BK,MinBKExp1,MinDiscount )

'[CALCULATE "BKBONUS" TO BE ADDED TO CUST FACT AS PART OF "FINETUNE"]
BKBonusCond = BPMod.bp_IFNE( vClass,5,1,0 ) * BPMod.bp_IFGE( TotalDown,Price *
0.20,1,0 ) * BPMod.bp_IFGE( TotalDown,1500,1,0 ) * BPMod.bp_IFGE( YrsTRW,5,1,0 ) *
BPMod.bp_IFG( TotalGood,5,1,0 )
BKBonusExp6 = BPMod.bp_IFGE( RealHiGood,10000,BKHiGood,0 )
BKBonusExp5 = BPMod.bp_IFB( Spouse,BKBonusExp6 + BKSpouse, BKBonusExp6 )
BKBonusExp4 = BPMod.bp_IFGE( BPMod.bp_IFB( Spouse,Inc + SpInc,Inc ),3000,BKBonusExp5
+ BKInc, BKBonusExp5 )
BKBonusExp3 = BPMod.bp_IFGE( TotalGood,8,BKBonusExp4 + BKGood, BKBonusExp4 )
BKBonusExp2 = BPMod.bp_IFE( MinBK,MinDiscStrongBK, BKBonusExp3 + BKStrong,
BKBonusExp3 )
BKBonusExp1 = BPMod.bp_IFB( BKBonusCond,BKBonusExp2, 0 )
BKBonus = BPMod.bp_MIN( BPMod.bp_IFB( BK, BKBonusExp1, 0 ), 1 )


'[DEBT MODEL #1 CALCULATE "COUNTRENT" AND "CRAPRATIO"]
OKCrap = BPMod.bp_IFB( Spouse, 0.18, 0.13 )
Crap = DEBT / RealInc
RentMult = ( Crap - CRStart ) / ( CRCountAll - CRStart )
CountRentExp2 = RentMult * Rent
CountRentExp1 = BPMod.bp_IFGE( Crap,CRCountAll,Rent,CountRentExp2 )
CountRent = BPMod.bp_IFG( Crap,CRStart,CountRentExp1,0 )
CrapRatio = BPMod.bp_MAX( Crap - OKCrap, 0 )
```

```
'[CALCULATE SIGNIFICANT DOWN = "SIGDOWN"]
Equity = BPMod.bp_MAX(( MaxCB - CB - WarrAllowance ), 0 )
DollarDownMult = BPMod.bp_MIN( RealDown, SDDollarDown ) / SDDollarDown
PercentDownMult = BPMod.bp_MIN( RealDown / Price, SDPercentDown ) / SDPercentDown
SigMult = BPMod.bp_MAX( BPMod.bp_MAX( DollarDownMult, PercentDownMult ),
SDEquityMult )
SigDown = BPMod.bp_MIN( SigMult * Equity * SDScaler, 0.5 * RealDown )


'[CALCULATE "DEBTADJUSTMENT"]
DAPmt = BPMod.bp_MAX( BPMod.bp_PMT(Interest, DATerm, CB, DaysToPay ), TooSmallPmt )
DebtAdjustment = BPMod.bp_IFNE( Term,DATerm,( DAPmt - Payment ) * DAScaler, 0 )

'[CALCULATE TOTAL DEBT = "TOTDEBT"]
MinDebt = BPMod.bp_MAX( BPMod.bp_MAX( CountRent, MinRent ), RealInc * 0.1 ) + Debt
InsDebt = BPMod.bp_IFB( BPMod.bp_IFE( Ins,0,1,0 ) * BPMod.bp_IFG( CB,2500,1,0 ), CB
* 0.01, 0 )
WarDebtExp = BPMod.bp_PMT( Interest, Term, WarrAllowance, DaysToPay )
WarDebt = BPMod.bp_IFG( WarrAllowance,0, WarDebtExp,0 )
TotDebt = MinDebt + Payment + InsDebt - WarDebt + DebtAdjustment


'[CALCULATE VARIABLE TIME ON JOB WHETHER MARRIED OR NOT = "REALJOB"]
JobInc = Job * ( Inc - Support )
SpJobInc = SpJob * SpInc
RealJobExp2 = ( JobInc + SpJobInc ) / TotalInc
RealJobExp1 = BPMod.bp_IFLE( TotDebt / ( Inc - Support ),0.40, BPMod.bp_MAX(Job,
RealJobExp2), RealJobExp2 )
RealJob = BPMod.bp_IFB( Spouse, RealJobExp1, Job )

'[CALCULATE "JOBTOT" TO BE USED IN CUST FACT DETERMINATION]
JobPoints1 = LookupJobTable( RealJob, 2 )
ExtraTime = RealJob - LookupJobTable ( RealJob, 1 )
JobPoints2 = LookupJobTable( RealJob, 3 ) * ExtraTime
JobTot = JobPoints1 + JobPoints2

'[CALCULATE BONUS POINTS FOR FTB OR SHORT BUREAU TO BE USED AS PART OF FINETUNE =
"SMALLFTBBONUS," "FTBBONUS"]
FTBPointsCond1 = BPMod.bp_IFLE( YrsTRW,1.1,1,0 ) * BPMod.bp_IFNE( vClass,5,1,0 ) *
BPMod.bp_IFE( Repos,0,1,0 ) * BPMod.bp_IFL( RealHiDerog,3000,1,0 )
FTBPoints7 = BPMod.bp_IFGE( RealInc,1500,FTBInc,0 )
FTBPoints6 = BPMod.bp_IFLE( Payment / RealInc,0.20,FTBPoints7 +
FTBPmtRatio,FTBPoints7 )
FTBPoints5 = BPMod.bp_IFLE( CB - Ins - SigDown,5500,FTBPoints6 + FTBCB,FTBPoints6 )
FTBPoints4 = BPMod.bp_IFB( PhBill,FTBPoints5 + FTBPhBill,FTBPoints5 )
FTBPoints3 = BPMod.bp_IFGE( (TotalDown / Price), 0.25, FTBPoints4 + 1 + ( TotalDown
/ Price - 0.25 ) / FTBDown,FTBPoints4 )
FTBPoints2 = BPMod.bp_IFB( Spouse,FTBPoints3 + FTBSpouse,FTBPoints3 )
FTBPoints1 = BPMod.bp_IFGE( Resid,2.1,FTBPoints2 + FTBResid,FTBPoints2 )
FTBPointsExp = BPMod.bp_IFGE( RealJob,2.1,FTBPoints1 + FTBJob,FTBPoints1 )
FTBPoints = BPMod.bp_IFB( FTBPointsCond1, FTBPointsExp, 0 )
SmallFTBBonus = ( 1.1 - YrsTRW ) * 0.25 * BPMod.bp_MIN ( 1, FTBPoints / 6 )
FTBBonus = BPMod.bp_IFG( FTBPoints,6, ( 1.1 - YrsTRW ) * 0.50 * BPMod.bp_MIN( 1, (
FTBPoints - 6 ) / 3 ), 0 )


'[BEGIN SPECIAL POINTS MODEL; YIELDS "FTSPECIALPOINTS"]
'[HIT FOR LOW JOB AND LOW RESID AT SAME TIME = "SHORTTIMEHIT"]
ShortTimeHitCond1 = BPMod.bp_IFLE( Job,STHit1,1,0 ) * BPMod.bp_IFLE(
Resid,STHit1,1,0 ) * BPMod.bp_IFB( Spouse, BPMod.bp_IFLE( SpJob,STHit1,1,0 ), 1 )
```

```
ShortTimeHitCond2 = BPMod.bp_IFLE( Job,STHit2,1,0 ) * BPMod.bp_IFLE(
Resid,STHit2,1,0 ) * BPMod.bp_IFB( Spouse, BPMod.bp_IFLE( SpJob,STHit2,1,0 ), 1 )
ShortTimeHit1 = ( ( STHit1 * STHit1 ) - ( Job * Resid ) ) * STScaler1
ShortTimeHitExp1 = BPMod.bp_IFB( ShortTimeHitCond2, ShortTimeHit1 + ( ( STHit2 *
STHit2 ) - ( Job * Resid ) ) * STScaler2, ShortTimeHit1 )
ShortTimeHit = BPMod.bp_IFB( ShortTimeHitCond1, ShortTimeHitExp1, 0 )

'[HIT FOR HI AMT FINANCED UNLESS OVERRIDE=Y; = "HICBHIT"]
HiCBNumber = BPMod.bp_IFG( ( CB - Ins - SigDown ), HCBAmtFin, ( CB - Ins - SigDown -
HCBAmtFin ) * HCBScaler, 0 )
HCO1 = BPMod.bp_IFE( ShortTimeHit,0, 1, 0 )
HCO2 = BPMod.bp_IFE( Repos,0, HCO1 + 1, HCO1 )
HCO3 = BPMod.bp_IFAND2( BPMod.bp_IFE( Repos,1,1,0 ), BPMod.bp_IFB( BK, 1,0 ), HCO2 +
1, HCO2 )
HCP1 = BPMod.bp_IFGE( TotalGood, TotalDerog, 1, 0 )
HCP2 = BPMod.bp_IFGE( RealHiGood, RealHiDerog, HCP1 + 1, HCP1 )
HCP3 = BPMod.bp_IFGE( RealHiGood, 0.50 * ( TotalLessIns - SigDown ), HCP2 + 1, HCP2
)
HCPExp = BPMod.bp_IFGE( HCP3, 2, 0, 1 )
HiCBOverideExp = BPMod.bp_IFGE( HCO3, 2, HCPExp, 1 )
HiCBOveride = BPMod.bp_IFL( HiCBNumber, 0, HiCBOverideExp,1 )
HiCBHit = HiCBNumber * HiCBOveride

'[EXTRA POINTS FOR OPTIMAL CB = "OPTIMALCBCREDIT"]
Variance = ABS( TotalLessIns - OptimalCB )
OptimalCBExp1 = BPMod.bp_IFB( BPMod.bp_IFGE( Payment, 240, 1,0 ) * BPMod.bp_IFE(
ShortTimeHit, 0, 1, 0 ) * BPMod.bp_IFGE( RealDown, 1000, 1, 0 ),( 1 - Variance /
AllowVariance ) * OptimalPoints, 0 )
OptimalCBCredit = BPMod.bp_IFL( Variance, AllowVariance, OptimalCBExp1, 0 )
FTSpecialPoints = OptimalCBCredit + HiCBHit + ShortTimeHit
'[END SPECIAL POINTS MODEL]



'[FINETUNE MODEL--TO BE ADDED TO CUST FACTOR = "FINETUNE"]
FTBonus = FTBBonus + SmallFTBBonus + BKBonus
FTPhBill = BPMod.bp_IFAND2( BPMod.bp_IFB( PhBill, 1, 0 ),BPMod.bp_IFL( TotalDerog +
TotalGood, 4, 1, 0 ), 0.12, 0 )
FTDerogHit = BPMod.bp_IFG( TotalDerog, 4, - 0.05 - 0.01 * ( TotalDerog - 5 ), 0 )
FTSigDown = SigDown * .0001 + BPMod.bp_IFG( SigDown, 2000, ( SigDown - 2000 ) *
.0001, 0 )
FTEquity = 0.75 - EquityTest + BPMod.bp_MAX( 0.6 - EquityTest, 0 )
FTBuyIFBreathing = BPMod.bp_MAX( FTSigDown, FTEquity ) * BPMod.bp_IFOR2(
BPMod.bp_IFL( TotalLessIns, ( MaxCB - Ins ) * 0.75, 1, 0 ), BPMod.bp_IFGE( SigDown,
1000, 1, 0 ), 1, 0 )
FTSmallHiDerog = BPMod.bp_IFB( BPMod.bp_IFE( TotalGood, 0, 1, 0 ) * BPMod.bp_IFLE(
RealHiDerog, 500, 1, 0 ) * BPMod.bp_IFG( YrsTRW, 1, 1, 0 ), 0.30, 0 )
FTBHD = BPMod.bp_IFB( BPMod.bp_IFGE( RealHiDerog, 2700, 1, 0 ) * BPMod.bp_IFB( BK,
0, 1 ) * BPMod.bp_IFE ( Repos, 0, 1, 0 ),( RealHiDerog / 8000 ) * - 0.60, 0 )
FTBigHiDerog = BPMod.bp_MAX( - 0.50,FTBHD )
InsCantFindErr = BPMod.bp_IFB( InsFlag, LookupIns( TotalLessIns,3 ), 0 )
FineTune = FTSpecialPoints + FTBigHiDerog + FTSmallHiDerog + FTBuyIFBreathing +
FTDerogHit + FTPhBill + FTBonus + InsCantFindErr



'[BEGIN FINAL CUSTOMER FACTOR CALCULATION -- ADD UP "F" VARIABLES]
TRWPart = BPMod.bp_IFL ( YrsTRW, 2, BPMod.bp_MIN( YrsTRW * 0.5, 0.9 ), BPMod.bp_MIN(
0.7 + YrsTRW * 0.1, 1 ) )
FTRW = TRWPart * CFTRWScaler
```

```
JobPart = JobTot / 10
FJob = JobPart * CFJobScaler

ResidPart = ResidTot / 10
FResid = ResidPart * CFResidScaler

GoodPart = BPMod.bp_IFL ( TotalGood, 2, TotalGood * 0.5, BPMod.bp_MIN( 0.5 +
TotalGood * 0.1, 1 ) )
FGood = GoodPart * GoodScaler

HiGoodPart = BPMod.bp_IFL ( RealHiGood, 20000, 0.5 * RealHiGood / 20000, 0.5 )
FHiGood = HiGoodPart * CFHiGoodScaler

DerogPart = BPMod.bp_IFL ( TotalDerog, 4, TotalDerog * - 0.25, - 0.5 - TotalDerog *
0.1 )
BKDerog = BPMod.bp_IFB ( BK, 0.7, 1 )
FDerog = BPMod.bp_MAX( DerogPart * BadScaler, - 1.05 ) * BKDerog

CFPhBillScaler = BPMod.bp_IFL( TotalLessIns, 4000, 0.8, 0.65 ) * 20 / Term
PhBillPart = BPMod.bp_IFB ( PhBill, BPMod.bp_IFL ( EquityTest, 0.90, 0.33, 0.33 *
0.80 ), 0 )
FPhBill = PhBillPart * CFPhBillScaler

RepoPart = Repos * - 0.25
CFRepoScaler = BPMod.bp_IFG( RealHiDerog, 1000, 2, BPMod.bp_MAX( 1, RealHiDerog *
.002 ) )
FRepo = RepoPart * CFRepoScaler

BKPart = BPMod.bp_IFB( BK, - 0.5, 0 )
FBK = BKPart * CFBKScaler

HomePart = BPMod.bp_IFB ( Home, 2/3, 0 )
HomePartScaler= 0.4 + 0.4*(BPMod.bp_IFG (RealHiGood, 30000, RealHiGood-30000,
0)/70000)
FHome = HomePart * BPMod.bp_MIN( CFHomeScaler, HomePartScaler)

IncPart = BPMod.bp_IFL ( RealInc, 3000, RealInc / 2000, BPMod.bp_MIN( RealInc,12000
) / 1800 )
FInc = IncPart * CFIncScaler

DebtPart = BPMod.bp_IFGE ( TotDebt / RealInc, 0.55, - 0.1, BPMod.bp_MIN( 0.7 -
TotDebt / RealInc, 0.5 ) )
FDebt = DebtPart * CFDebtScaler

CFSpouseScaler = BPMod.bp_IFLE( YrsTRW, 1, 0.5, 0.35 )
WorthlessSpouse = BPMod.bp_IFAND2( BPMod.bp_IFLE( SpJob, 0, 1, 0 ),BPMod.bp_IFLE(
SpGood, 0, 1, 0 ),0, 1 )
SpousePart = BPMod.bp_IFB ( Spouse, 0.5, 0 ) * WorthlessSpouse
FSpouse = SpousePart * CFSpouseScaler

CoxPart = BPMod.bp_IFB ( Cox, 0.5, 0 )
CFCoxScaler = CoxScaler / 10
FCox = CoxPart * CFCoxScaler


TotalCFPoints = FTRW + FJob + FResid + FGood + FHiGood + FDerog + FPhBill + FRepo +
FBK + FHome + FInc + FDebt + FSpouse + FCox + FineTune
CustFact = BpMod.bp_ROUND(BPMod.bp_MAX( BPMod.bp_MIN(TotalCFPoints, 5 ), 0.001 ) *
0.98, 2)
'[END FINAL CUSTOMER FACTOR CALCULATION]
```

```
'[CALCULATE SCALER IF GOOD CUSTOMER WITH HIGH DEBT = "DEBTSCALER"]
DebtScaler_exp1 = BPMod.bp_IFOR2( BPMod.bp_IFLE( TotalLessIns,RealInc * 5,1,0
),BPMod.bp_IFLE( TotalLessIns,4500,1,0 ),1,0 )
DebtScaler_exp2 = BPMod.bp_IFGE( RealJob,1,1,0 )
DebtScaler_exp3 = BPMod.bp_IFGE( YrsTRW,1,1,0 )
DebtScaler_exp4 = BPMod.bp_IFOR2( BPMod.bp_IFLE( TotalDerog,1,1,0 ),BPMod.bp_IFLE(
RealHiDerog,400,1,0 ),1,0 )
DebtScaler_exp5 = BPMod.bp_IFLE( TotalGood,4,1,0 )
DebtScaler_exp6 = BPMod.bp_IFL( TotalGood,TotalDerog,1,0 )
DebtScaler_exp7 = BPMod.bp_IFL( RealInc,1700,1,0 )
DebtScalerCondition = DebtScaler_exp1 * DebtScaler_exp2 * DebtScaler_exp3 *
DebtScaler_exp4 * DebtScaler_exp5 * DebtScaler_exp6 * DebtScaler_exp7
DSInc = BPMod.bp_MAX( RealInc, 1200 )
DebScalerExp = 0.5 + ( DSInc - 1200 ) / 1000
DebtScaler = BPMod.bp_IFB( DebtScalerCondition,DebScalerExp,1 )


'[CALCULATE DEBT RATIO HIT FOR PAY PROB ADJUSTMENTS = "DEBTPROBLEM"]
DebtRatio = RealInc / TotDebt
DebtHitExp = BPMod.bp_IFLE( DebtRatio,2, 0.225 + ( 2 - DebtRatio ) * 0.6, ( 2.5 -
DebtRatio ) * 0.45 )
DebtHit = BPMod.bp_IFLE( DebtRatio, 2.5, DebtHitExp,0 )
DHMax1 = BPMod.bp_MAX( 0.95 - EquityTest, 0 )
DHMax2 = BPMod.bp_MAX( 0.75 - EquityTest, 0 )
DebtHitScaler = 1.05 - DHMax1 - DHMax2
DebtProblem = DebtHit * DebtHitScaler * DebtScaler


'[EXCESS TERM DETERMINATION MODEL BEYOND BASETERM--MODEL YIELDS "FREETERM",
"BUYTERM", & "EXTERM"]
BaseTerm = 31
MinPmt = 255 - ( SigDown / 75 )
OKPmt = BPMod.bp_IFGE( Payment,MinPmt,1,0 )

RegularFreeTerm = BPMod.bp_IFG( CustFact, FreeGetNone, 1, 0 )

YEMiles = LookupTermTable( vClass, 2 )
YEAge = LookupTermTable( vClass, 3 )
MEAge = LookupTermTable( vClass, 4 )
MEMiles = LookupTermTable( vClass, 5 )

FreeTermPercent = BPMod.bp_MIN( ( CustFact - FreeGetNone ) / ( FreeGetAll -
FreeGetNone ), 1 )
Term4NewerCar = BPMod.bp_IFLE( Miles, YEMiles, BPMod.bp_MAX( YEAge - CarAge, 0 ), 0
) * FreeTermPercent
Term4LowMiCar = BPMod.bp_IFLE( CarAge, MEAge, BPMod.bp_MAX( ( MEMiles - Miles ) /
5000, 0 ), 0 ) * FreeTermPercent
StrongBuyerFreeTerm = BPMod.bp_IFG ( CustFact, SBGetNone, 1, 0 )

SBAge = LookupTermTable( vClass, 6 )
SBMiles = LookupTermTable( vClass, 7 )

SBFreeTermPercent = BPMod.bp_MIN( ( CustFact - SBGetNone ) / ( SBGetAll - SBGetNone
), 1 )
Term4StrongBuyer = BPMod.bp_IFAND2( BPMod.bp_IFLE ( CarAge, SBAge, 1, 0
),BPMod.bp_IFLE ( Miles, SBMiles, 1, 0 ),3 * SBFreeTermPercent, 0 )
QualifyFreeTerm = BPMod.bp_IFAND2( BPMod.bp_IFB( RegularFreeTerm, 1, 0
),BPMod.bp_IFB( StrongBuyerFreeTerm, 1, 0 ),Term4NewerCar + Term4LowMiCar +
Term4StrongBuyer,BPMod.bp_IFB ( RegularFreeTerm, Term4NewerCar + Term4LowMiCar, 0 )
)
FreeTerm = BPMod.bp_IFG( Term, BaseTerm, BPMod.bp_MIN( QualifyFreeTerm, Term -
BaseTerm ), 0 ) * OKPmt

OKTerm = BaseTerm + FreeTerm
BuyTerm = BPMod.bp_MAX( Term - OKTerm, 0 )

ExTermScaler = ( CustFact - 1 ) / 1.75 * 0.01
ExcessCharge = BPMod.bp_IFG ( CustFact, 1, 0.015 - ExTermScaler, 0.015 )
CostPerMonth = BPMod.bp_MAX( ExcessCharge, .005 )
PmtBelow250 = BPMod.bp_IFL ( Payment, 250, 1000, 1 )
TooLong = BPMod.bp_IFG ( BuyTerm, 6, 1000, 1 )
MustBuyTerm = BPMod.bp_IFGE ( BuyTerm, 0, 1, 0 )
ExTerm = BPMod.bp_IFB( MustBuyTerm, CostPerMonth * BuyTerm * PmtBelow250 * TooLong,
0 ) * (CB - Ins - WarrAllowance)
'[END EXCESS TERM DETERMINATION MODEL]


'[PRIMARY TERM HIT/HELPER = "XTERM"]
TermCust = CustFact * 20
KentTerm = ( 12 - CarAge ) * 6
ClassTerm = 5 - vClass
ClassScaler = ClassTerm / 5
CBTerm = BPMod.bp_IFG ( ( MaxCB - Ins ), 6000, ( MaxCB - Ins - 6000 ) / 500, 0 )
TermCFScaler = BPMod.bp_IFG ( CustFact, 1, BPMod.bp_MIN( CustFact - 1, 1 ), 0 )
TermCar = KentTerm + ( ClassTerm + CBTerm * ClassScaler ) * TermCFScaler
TermMaxMiles = 180000 - ( vClass * 10000 )
SubtractTerm = BPMod.bp_IFG( Miles, TermMaxMiles, ( ( Miles - TermMaxMiles ) / 10000
) * vClass / 2, 0 )
TermMax = BPMod.bp_MIN( TermCar, TermCust ) + BuyTerm * 0.5 + FreeTerm * 0.5 -
SubtractTerm
XTerm = Term - TermMax


'[PAYMENT PROBABILITY MODEL]
'[CUSTOMER FACTOR COMPONENT = "CFALLOWANCE"]
CFSMin = LookupCFScalerTable( CustFact, 1 )
CFSBase = LookupCFScalerTable( CustFact, 2 )
CFSExtra = LookupCFScalerTable( CustFact, 3 )
CustFactScaler = CFSBase + ( CustFact - CFSMin ) * CFSExtra
CFAllowance = CustFactScaler * CustFact

'[DOWN PAYMENT COMPONENT = "DOWNPRICE"]
FedExTax = BPMod.bp_IFGE( CustFact, 2.5, 0, BPMod.bp_MIN( ( 2.5 - CustFact ) * 76,
39 ) )
InputDiscount = Reserve - FedExTax
DownAllowance = ( Price * 0.2 ) + InputDiscount + SigDown - ExTerm
DownPrice = DownAllowance / Price

'[OVERALL SCALER]
PPScaler = 0.95

'[ADJUSTMENTS = "PPADJUST"]
PPDebt = DebtProblem * - 0.7
PPCrap = ( CrapRatio * DebtScaler ) * - 1
PPStupid = BPMod.bp_IFB( BPMod.bp_IFL( Payment/Term, StupidNum, 1, 0 ) *
BPMod.bp_IFGE( Term,StupidTerm,1,0 ), ( StupidNum - Payment / Term ) * - 0.1, 0 )
PPTerm = XTerm * - 0.01
PPAdjust = PPTerm + PPDebt + PPStupid + PPCrap

PayProb = CFAllowance * DownPrice * PPScaler + PPAdjust
'[END PAYMENT PROBABILITY MODEL]
```

```
'[DISCOUNT NEEDED BASED ON PAYMENT PROBABILITY MODEL = "SPREADNUM"]
LossProb = BPMod.bp_MIN ( 1 - PayProb, 1.1 )
DiscountAllow = InputDiscount * 2
LossAmount = LossProb * ( TotalLessIns - WarrAllowance ) + ExTerm - DiscountAllow
Spread = SpreadReq * ( CB - Ins - WarrAllowance )
SpreadNum = ( LossAmount + Spread ) * SpreadNumScaler



'[MINIMUM % DISCOUNT AREA]
'[CALCULATE MIN % DISCOUNT DEPENDING OF # REPOS = "MINREPO"]
MinRepoExp3 = BPMod.bp_CASE3( repos, 1, 2, 3, 0.125, 0.20, 0.35, 0.50 )
MinRepoExp2 = BPMod.bp_CASE2( repos, 1, 2, 0.10, 0.175, 0.30 )
MinRepoExp1 = BPMod.bp_IFB( BK,MinRepoExp2, MinRepoExp3 )
MinRepo = BPMod.bp_IFE( Repos,0,0.10,MinRepoExp1 )

'[CALCULATE MIN % DISCOUNT BASED ON HI DEROG = "MINDEROG"]
MinDerog = BPMod.bp_IFB( BPMod.bp_IFGE( RealHiDerog,3000,1,0 ) * BPMod.bp_IFE( BK,0,
1,0 ), MinDiscHiDerog, MinDiscount )

'[CALCULATE MIN % DISCOUNT BASED ON LOW TIME ON BUREAU = "MINTRW"]
MinTRWExp = BPMod.bp_IFOR2( BPMod.bp_IFGE( FTBPoints,9,1,0 ), BPMod.bp_IFGE(
CoxScaler,30,1,0 ), 0.125 - ( YrsTRW / 40 ), 0.15 - ( YrsTRW / 20 ) )
MinTRW = BPMod.bp_IFL( YrsTRW, 1, MinTRWExp , 0.10 )

'[CALCULATE MIN % DISCOUNT BASED ON CUSTOMER FACTOR = "MINFACT"]
FactMinDisc = 0.3
SigDownHelper = ( SigDown * 0.25 ) / ( TotalLessIns - WarrAllowance )
Below75 = BPMod.bp_IFL( CustFact, 0.75, 1, 0 )
Below75Hit = BPMod.bp_IFL( CustFact, 0.35, .2, ( 75 - ( CustFact * 100 ) ) * .005 )
Below35 = BPMod.bp_IFL( CustFact, 0.35, 1, 0 )
Below20 = BPMod.bp_IFL( CustFact, 0.20, 1, 0 )
LowBalScaler = BPMod.bp_IFLE( TotalLessIns, 2000, 0.50, BPMod.bp_IFLE( TotalLessIns,
3000, 1 - ( ( 3000 - TotalLessIns ) / 1000 ) * 0.50, 1 ) )
MinFact75 = ( FactMinDisc + Below75Hit - SigDownHelper ) * LowBalScaler
MinFact35 = BPMod.bp_IFB( Below35, BPMod.bp_IFB( Below20, 10, BPMod.bp_IFG(
TotalLessIns, 3000, 10, 0 ) ), 0 )
MinFact = BPMod.bp_IFB( Below75, BPMod.bp_MAX( MinFact75, MinFact35 ), MinDiscount )
'[END MINIMUM % DISCOUNT AREA]



'[ADDITIONAL DISCOUNT FOR KINKY TERM = "KINKTERM"]
TermIsKinky = BPMod.bp_IFG( Term, KinkMaxTerm, 1, 0 )
KinkSubtot = BPMod.bp_MAX( CarAge - KinkAge, 0 ) + BPMod.bp_MAX( Miles - KinkMiles,
0 ) / 10000
PointsFromCF = BPMod.bp_MAX( KinkCF - CustFact, 0 ) * 10 * KinkSubTot
OverMax = BPMod.bp_MIN( BPMod.bp_MAX( Term - KinkMaxTerm, 0 ), 3 )
TotalKinkPoints = ( KinkSubtot * OverMax ) + ( KinkSubtot * PointsFromCF * OverMax )
KinkTerm = BPMod.bp_IFB( TermIsKinky, TotalKinkPoints * CostPerKinkPoint, 0 )



'[GET FINAL RESERVE]
MinDisc = BPMod.bp_IFGE( CustFact, 2.5, 300, BPMod.bp_MIN( ( 2.5 - CustFact ) * 88 +
300, 344 ) )
MinPercent = BPMod.bp_MAX( BPMod.bp_MAX( BPMod.bp_MAX( MinDerog, MinTRW ),
BPMod.bp_MAX( MinBK, MinRepo ) ), BPMod.bp_MAX( MinFact, MinDiscount ) )
MinReserve = MinPercent * ( TotalLessIns - WarrAllowance )
FinalSubtot = BPMod.bp_MAX( BPMod.bp_MAX( MinDisc, MinReserve ), SpreadNum )
TooMuchTerm = BPMod.bp_IFG( Term, 48, 50000, 0 )
```

```
PmtTooSmall = BPMod.bp_IFL( Payment, TooSmallPmt, 50000, 0 )

FinalReserve = FinalSubtot + KinkTerm + ExTerm + TooMuchTerm + PmtTooSmall


'[GET OVERADVANCE AND CHECK TO DEALER]
REALOA=BPMod.bp_IFG(CB, MAXCB, CB-MAXCB, 0.00)
CheckToDealer=CB-INS-RESERVE-ACQFEE-REALOA
OA=Round(REALOA+0.50, 0)



'[HINT AND ERROR SECTION]
'[NEED THE FOLLOWING TO BEGIN HINTS]
DebtP= TotDebt/RealInc
DebtDiff= DebtP - 0.55
LessDebt= DebtDiff*RealInc + 5
GetDown= (2000-RealDown)*0.8
LowerPrice= (1000-SigDown-GetDown)/0.8


'[HINTS]
'CHANGE
if repos > 3 then
hint1 = " Wow! " + formatnumber(repos,0) + " repossessions!!!  "
end if

'CHANGE
if repos > 2 then
hint2 = "  But...  " + formatnumber(repos,0) + " repossessions?  Forget the phone
bill, get a blood sample.  "
end if

'CHANGE
if ((PPStupid+PPTerm < -0.15) and (FinalReserve > (CB-Ins)*0.15) and (FinalReserve >
500)) then
hint3 = " You could do better with a shorter term.  "
end if

'CHANGE
if ((DealerGross < 0) and (RealInc < 1400)) then
hint4= " Try a less expensive car for this income so you can make a better deal."
end if

'CHANGE
if ((CustFact < 0.75) and (MinRepo*(TotalLessIns-WarrAllowance) <=
FinalReserve-200)) then
hint5= " Try a lower price, or more down, or a shorter term.   You might make a
better deal.  "
end if

'CHANGE
if ((YrsTRW = 0) and (Good > 0) and (Good < 3)) then
hint6= " Make sure you get documentation showing the good credit.  No rental,
medical, or dental. "
end if

'CHANGE
if ((Home = 1) and (HiGood < 30000)) then
hint7= " If the house is not on the credit bureau, then make sure to send proof of
home-owner.  "
end if
```

```
'CHANGE
'if ((Miles <> 117545) or (Price <> 6995)) then
if (InputDiscount >= FinalReserve) then
hint8= " It's a deal!     "
end if

'CHANGE
if (Miles < 100000 and (BPMod.bp_THISYEAR-CarYear > 9)) then
hint9= " Better check the miles.  If the car is over 10 years old, you have to input
at least 100,000 miles."
end if

'CHANGE
if Miles < (BPMod.bp_THISYEAR-CarYear)*7000 then
hint10= " Check your miles.  Your input is very low, unless the last owner was my
grandmother."
end if

'CHANGE
if repos >= 5 then
hint11= "  Like a '72 Pinto.  "
end if

'CHANGE
if ((Repos > 0) and (HiDerog < 3000)) then
hint12= " Don't forget that the Hi Derog is the amount of the loan, not how much was
charged off.  "
end if

'CHANGE
if BK = 1 then
hint13= " Bankruptcy must be discharged.  "
end if

'CHANGE
if ((YrsTRW = 0) and (Good > 2)) then
hint14= " You can't have more than 2 good credit items that are not on the bureau.
"
end if

'CHANGE
if ((Job > 2) and (Resid > 2) and (Job = Resid)) then
hint15= " If this is a military deal, don't forget to send a completed Mac
allotment.  Must be rank of E3 or higher.  "
end if

'CHANGE
if Support > 0 then
hint16= "  Remember not to count Family Support accounts as Good or Derog. "
end if

'CHANGE
if ((DebtPart < 0) and (LessDebt < 40) and (FinalReserve >= BPMod.bp_MAX(MinDisc,
MinReserve) + KinkTerm + ExTerm + 100) and (Payment-LessDebt > 170)) then
hint17= " You can make a better deal if you use Price and Down to get the payment
about " + formatnumber(LessDebt,0) + " dollars lower.  "
end if

'CHANGE
if ((DebtPart < 0) and (LessDebt > 40) and (FinalReserve >= BPMod.bp_MAX(MinDisc,
MinReserve) + KinkTerm + ExTerm + 100) and (Payment-LessDebt > 170)) then
hint18= "  You could make a lot better deal if the payment was " +
formatnumber(LessDebt,0) + " dollars lower.  Try a less expensive car.  "
```

```
end if

'CHANGE
if GetDown+SigDown <= 1000 then
hint19= " And lower the Price by about " + formatnumber(LowerPrice, 0) + " dollars.
"
end if

'CHANGE
if ((SigDown >= 850) and (SigDown < 1000) and (FinalReserve >= BPMod.bp_MAX(MinDisc,
MinReserve) + KinkTerm + ExTerm + 100)) then
  if RealDown < 2000 then
    hint20= "  You might do better if you get 2000 dollars down. " +hint19
  else
    hint20= "  You might do better if you lower the Price by about " +
formatnumber((1000-SigDown)/0.8, 0) + " dollars.  "
  end if
end if

'CHANGE
hint22= " Try putting down  " + dollarString(OA,0) + "  more, or lower the price."

if ((CustFact > 1.0000000000) and (OA > 0)) then
  hint21= " Try putting down  " + formatnumber(OA,0) + " dollars more, or reserve
the O-A, then:"
  else if (OA > 0) then
    hint21= ""
    hint25=hint22
    hint8= ""
  end if
end if

if (CustFact < 1.000000000) then
  hint23= " You can't reserve the O-A, because the Customer Factor has to be over 1.
"
else
  hint23= ""
end if

'if (vClass = 5) then
'  hint24= " You can't reserve the O-A, because the Car Class cannot be 5. "
'else
'  hint24= ""
'end if


'[ERROR CHECKING]
DeathErr4 = BPMod.bp_IFG( KinkTerm,( CB - Ins ) * .1, 4, 0 )
DeathErr3 = BPMod.bp_IFL( Payment, TooSmallPmt, 3, DeathErr4 )
DeathErr2 = BPMod.bp_IFGE( ExTerm, 0.25 * ( CB - Ins ), 2, DeathErr3 )
DeathErr1 = BPMod.bp_IFG( Term, 48, 1, DeathErr2 )

Err9 = BPMod.bp_IFL( FCox,0,9,10 )
Err8 = BPMod.bp_IFG( KinkTerm,( CB - Ins ) * .02, 8, Err9 )

if (CB-MAXCB <= 0.00) then
  Err7 = Err8
  else
  if (CB-MAXCB < 300) then
    Err7 = 12
    else
    if (CB-MAXCB <= 1000) then ' [o/a is between 300 and 1000, inclusive]
      Err7 = 7
      else
      Err7 = 11
    end if
  end if
end if

Err6 = BPMod.bp_IFL( Reserve, 300, 6, Err7 )
Err5 = BPMod.bp_IFL( Reserve, .10 * ( CB - Ins - WarrAllowance ), 5, Err6 )


ErrCode = BPMod.bp_IFG( FinalReserve, CB, DeathErr1, Err5 )
Errstr = ErrLookup(ErrCode)
NoDollarOA=FormatNumber(REALOA,0)

if (REALOA = 0.00) then
  OAStr = ""
else
  OAStr = "$ " & FormatNumber(OA,0)
end if

StructOK  = InputDiscount >= FinalReserve
AMTOK = CB <= MAXCB

Set BPMod = Nothing

If Err.Number <> 0 then
  SystemError = Err.Description
End if

'[END EXPRESSIONS]
```

```
None,0
Aladdin Resort & Casino,1
Arizona Charlies East,4
Aztec Inn Casino,4
Arizona Charlies,3
Barbary Coast,3
Bally's Las Vegas,1
Bellagio,1
Binion's Horseshoe,5
Boardwalk Casino,3
Boulder Station Hotel & Casino,2
Caesars Palace,1
California Hotel & Casino,2
Castaways Hotel,3
Circus Circus Hotel & Casino,2
Crowne Plaza,5
Excalibur Hotel & Casino,1
El Cortez Hotel & Casino,5
Fiesta Casion Hotel,4
Flamingo Las Vegas,1
Fitzgeralds Casino & Holiday,1
Four Queens Casino / Hotel,2
Fremont Hotel & Casino,2
Golden Nugget,1
Gold Coast Hotel & Casino,3
Hacienda Hotel & Casino,3
Hampton Inn Tropicana,4
Hard Rock Hotel & Casino,1
Harrah's Las Vegas Casino,2
Hotel San Remo,2
Hyatt Regency Lake Las Vegas,2
Imperial Palace Hotel & Casino,2
Key Largo Casino & Quality Inn,4
Lady Luck Casino & Hotel,3
Las Vegas Club,2
Las Vegas Hilton,1
Luxor Las Vegas,1
Main Street Station,3
Mandalay Bay Resort & Casino,1
Maxim Hotel & Casino ,4
MGM Grand Hotel & Casino,1
Mirage Hotel & Casino,1
Monte Carlo Resort & Casino,1
Nevada Palace Hotel & Casino,3
New Frontier,2
New York - New York,1
Orleans Hotel & Casino,2
Palace Station Hotel & Casino,2
Palms Casino Hotel,2
Paris Las Vegas Casino Resort,1
Plaza Hotel & Casino,3
Regent Las Vegas,3
Reserve Hotel & Casino,3
Rio All Suite Hotel & Casino,1
Riviera Hotel & Casino,1
Sahara Hotel & Casino,1
Sam's Town Hotel & Gambling,1
Santa Fe Station & Hotel,1
Silverton Hotel & Casino,4
Stardust Resort & Casino,2
Stratosphere Casino Hotel,2
Suncoast Hotel & Casino,4
Sunset Station Hotel & Casino,2
```

20050107 ' 9/954001

```
Terrible's Hotel & Casino,4
Texas Station Gambling Hall,2
Treasure Island,1
Tropicana Resort & Casino,1
Venetian Resort Hotel,1
Union Plaza Hotel & Casino,3
Vacation Village,3
Westward Ho Hotel & Casino,3
Wild Wild West Hotel,3
ALL OTHER CASINOS,1
Strip Clubs (Click one below):,0
Can Can Room,6
Cheetah,6
Club Paradise,6
Crazy Horse Too,6
déjà vu,6
Diva's,7
Forbidden Club,7
Library,7
Lil' Darlings,7
Olympic Gardens,6
Spearmint Rhino,6
Strip Tease,7
Talk of the Town,7
All OTHER STRIP CLUBS,8
```

```
None/Not Full Time,0
Bar Back,4
Bartender,1
Bell Person,5
Change Person,4
Cocktail Server,2
Dealer,1
Exotic Dancer,9
Food Server,3
Room Service,4
Valet,5
Other Tipped Employee,4
```

```
unit main;

interface

uses
  ActiveX, MtsObj, Mtx, ComObj, BPfunctionsModule_TLB, StdVcl;

type
  TBPFunctions = class(TMtsAutoObject, IBPFunctions)
  protected
    function bp_IFG(Value1: Double; Value2: Double; Result1: Double; Result2: Double):
      Double; safecall;
    function bp_IFB(Condition, Result1, Result2: Double): Double; safecall;
    function bp_IFL(Value1, Value2, Result1, Result2: Double): Double;
      safecall;
    function bp_IFNE(Value1, Value2, Result1, Result2: Double): Double;
      safecall;
    function bp_CASE2(Key, Value1, Value2, Result1, Result2,
      ResultElse: Double): Double; safecall;
    function bp_CASE3(Key, Value1, Value2, Value3, Result1, Result2, Result3,
      ResultElse: Double): Double; safecall;
    function bp_CASE4(Key, Value1, Value2, Value3, Value4, Result1, Result2,
      Result3, Result4, ResultElse: Double): Double; safecall;
    function bp_IFAND2(Value1, Value2, Result1, Result2: Double): Double;
      safecall;
    function bp_IFE(Value1, Value2, Result1, Result2: Double): Double;
      safecall;
    function bp_IFLE(Value1, Value2, Result1, Result2: Double): Double;
      safecall;
    function bp_IFOR2(Value1, Value2, Result1, Result2: Double): Double;
      safecall;
    function bp_MAX(Value1, Value2: Double): Double; safecall;
    function bp_MIN(Value1, Value2: Double): Double; safecall;
    function bp_OCCAPR(LoanDate, FirstPaymentDate: TDateTime; InterestRate,
      Principal, Term, Payment: Double): Double; safecall;
    function bp_ROUND(NumberToRound, Exponent: Double): Double; safecall;
    function bp_TRUNC(NumberToTrunc, Exponent: Double): Double; safecall;
    function bp_IFGE(Value1, Value2, Result1, Result2: Double): Double;
      safecall;
    function bp_ADDONPMT(Principal, Term, AddOnRate,
      DaysToFirstPayment: Double): Double; safecall;
    function bp_PMT(InterestRate, Term, Principal,
      DaysToFirstPayment: Double): Double; safecall;
    function bp_VEHICLEAGE(VehicleYear, MonthOfManufacture: Double): Double;
      safecall;
    function bp_CEILING(NumberToCeiling, Exponent: Double): Double; safecall;
    function bp_FLOOR(NumberToFloor, Exponent: Double): Double; safecall;
    function bp_THISYEAR: Double; safecall;
    { Protected declarations }
  end;

const OccDaysInYear = 360;

implementation

uses ComServ, math, StDate, SysUtils;

function IncMonth(const date:TDateTime):TDateTime;
var
  orpDate : TstDate; { Orpheus functions }
begin
  orpDate := STDate.DateTimeToSTDate(date);
  orpDate := IncDateTrunc(orpdate, 1, 0);
  result := STDate.STDateToDateTime(orpDate);
end;

function DecMonth(const date:TDateTime):TDateTime;
var
  orpDate : TstDate; { Orpheus functions }
begin
  orpDate := STDate.DateTimeToSTDate(date);
  orpDate := IncDateTrunc(orpdate, -1, 0);
  result := STDate.STDateToDateTime(orpDate);
end;

function ComputeOccUnitPeriods(const firstPaymentDate:TDateTime;
                               const loanDate:TDateTime;
                               var   unitPeriods:extended ):extended;
var
```

```
  D2 : TDateTime;
  D1 : TDateTime;
  done : boolean;
begin
  D2 := firstPaymentDate;
  unitPeriods := 0;
  done := False;
  repeat
    D1 := D2;
    D2 := DecMonth(D2);
    if D2 >= loanDate then
      unitPeriods := unitPeriods + 1
    else
      done := True;
  until done;
  result := Trunc(D1 - loanDate);
end;



function TBPFunctions.bp_IFG(Value1, Value2 , Result1,
  Result2: Double): Double;
begin
  if (value1 > value2) then
    result := result1
  else
    result := result2
end;

function TBPFunctions.bp_IFB(Condition, Result1, Result2: Double): Double;
begin
  if (Condition = 0) then
    result := result2
  else
    result := result1
end;

function TBPFunctions.bp_IFL(Value1, Value2, Result1,
  Result2: Double): Double;
begin
  if (value1 < value2) then
    result := result1
  else
    result := result2
end;

function TBPFunctions.bp_IFNE(Value1, Value2, Result1,
  Result2: Double): Double;
begin
  if (value1 <> value2) then
    result := result1
  else
    result := result2
end;

function TBPFunctions.bp_CASE2(Key, Value1, Value2, Result1, Result2,
  ResultElse: Double): Double;
begin
  if (key = Value1) then
    result := Result1
  else if (key = Value2) then
    result := Result2
  else
    result := ResultElse;
end;

function TBPFunctions.bp_CASE3(Key, Value1, Value2, Value3, Result1,
  Result2, Result3, ResultElse: Double): Double;
begin
  if (key = Value1) then
    result := Result1
  else if (key = Value2) then
    result := Result2
  else if (key = Value3) then
    result := Result3
  else
    result := ResultElse;
end;

```

```
function TBPFunctions.bp_CASE4(Key, Value1, Value2, Value3, Value4,
  Result1, Result2, Result3, Result4, ResultElse: Double): Double;
begin
  if (key = Value1) then
    result := Result1
  else if (key = Value2) then
    result := Result2
  else if (key = Value3) then
    result := Result3
  else if (key = Value4) then
    result := Result4
  else
    result := ResultElse;
end;

function TBPFunctions.bp_IFAND2(Value1, Value2, Result1,
  Result2: Double): Double;
begin
  if (value1 >= 1) and (value2 >= 1) then
    result := result1
  else
    result := result2
end;

function TBPFunctions.bp_IFE(Value1, Value2, Result1,
  Result2: Double): Double;
begin
  if (value1 = value2) then
    result := result1
  else
    result := result2
end;

function TBPFunctions.bp_IFLE(Value1, Value2, Result1,
  Result2: Double): Double;
begin
  if (value1 <= value2) then
    result := result1
  else
    result := result2
end;

function TBPFunctions.bp_IFOR2(Value1, Value2, Result1,
  Result2: Double): Double;
begin
  if (value1 >= 1) or (value2 >= 1) then
    result := result1
  else
    result := result2
end;

function TBPFunctions.bp_MAX(Value1, Value2: Double): Double;
begin
  if Value1 > Value2 then
    Result := Value1
  else
    Result := Value2;
end;

function TBPFunctions.bp_MIN(Value1, Value2: Double): Double;
begin
  if Value1 < Value2 then
    Result := Value1
  else
    Result := Value2;
end;

function TBPFunctions.bp_OCCAPR(LoanDate, FirstPaymentDate: TDateTime;
  InterestRate, Principal, Term, Payment: Double): Double;
var
  done : boolean;
  U1   : double;
  R    : double;
  a,
  apr,
  p1   : Double;
  P0,
  X,
  U,
  P,        { amount financed }
  APU : double;  { actual APR }

  procedure ComputePV;
  var
    x1 : extended;
    v1 : extended;
    v2 : extended;
    v3 : extended;
    z  : extended;
    Y  : extended;
    occOddDays : extended;
    unitPeriods : extended;
  begin
    occOddDays := ComputeOccUnitPeriods(firstPaymentDate, loanDate, unitPeriods);
    P0 := 0.00;
    X1 := 1.00 + X;
    Y  := 1.00 + OccOddDays * X / U;
    V1 := (1.00 / power(X1, unitPeriods)) / Y;
    V2 := 1.00 / power(X1, term);
    V3 := 1.00 - V2;
    Z  := V1 * X1 * payment * V3 / X;
    P0 := P0 + Z;
  end;
begin

  try
    P := Principal;
    U := 30;
    U1 := U / OccDaysInYear;
    R  := InterestRate * U1 / 100.00;
    done := False;
    while not done do
    begin
      X := 0.0001;
      IF R <> 0.000 THEN
        X := R;

      ComputePV;
      P1 := P0;
      X  := R + 0.0001;
      ComputePV;

      APR := R + (P - P1) / (P0 - P1) * 0.0001;

      A  := ABS(APR - R);
      IF A > 0.0000001 THEN
        R := APR
      else begin
        APU := (100.00 / U1) * APR;
        done := True;
      end
    end; { while }
    result := APU;
  except
    result := 0.00;
  end;
end;



function TBPFunctions.bp_ROUND(NumberToRound, Exponent: Double): Double;
var
  TempNum1 : extended;
  TempNum2 : int64;
begin
  TempNum1 := NumberToRound * power(10, Exponent);
  TempNum2 := round(TempNum1);
  result := TempNum2 * power(10, (Exponent * (-1)));
end;

function TBPFunctions.bp_TRUNC(NumberToTrunc, Exponent: Double): Double;
var
  TempNum1 : extended;
  TempNum2 : int64;
begin
  TempNum1 := NumberToTrunc * power(10, Exponent);
  TempNum2 := trunc(TempNum1);
  result := TempNum2 * power(10, (Exponent * (-1)));
```

```
end;

function TBPFunctions.bp_IFGE(Value1, Value2, Result1,
  Result2: Double): Double;
begin
  if (value1 >= value2) then
    result := result1
  else
    result := result2
end;

function TBPFunctions.bp_ADDONPMT(Principal, Term, AddOnRate,
  DaysToFirstPayment: Double): Double;
var
  netTerm : extended;
  oddTerm : extended;
  oddDay : extended;
  FinanceCharge : extended;
begin
  oddDay := daysToFirstPayment - 30;
  oddTerm := oddDay / 30;
  netTerm := term + oddTerm;

  if addOnRate < 1.00 then
    financeCharge := principal * addOnRate * netTerm / 12
  else
    financeCharge := principal * addOnRate * 0.01 * netTerm / 12;

  result := (principal + financeCharge)/term;
end;

function TBPFunctions.bp_PMT(InterestRate, Term, Principal,
  DaysToFirstPayment: Double): Double;
var
  apr: extended;
  pv : extended;
  //fv : extended;
  oddDay : extended;
  dailyCharge : extended;
  oddDayCharge : extended;
begin
  if interestRate > 1.00 then
    apr := interestRate/100.00
  else
    apr := interestRate;

  oddDay := (daysToFirstPayment - 30);
  dailyCharge := (principal * apr)/365;
  oddDayCharge := oddDay * dailyCharge;
  pv   := principal + oddDayCharge;
  result := - Math.Payment(apr/12, trunc(term), pv, 0, ptEndOfPeriod);
end;

function TBPFunctions.bp_VEHICLEAGE(VehicleYear,
  MonthOfManufacture: Double): Double;
var
   PastYear,NumberOfMonths,NumberOfYears,PresentYear,PresentMonth,Day : word;
begin
   try
      DecodeDate(Date,PresentYear,PresentMonth,Day);
      PastYear := trunc(VehicleYear);
      if (PresentYear > PastYear -1) then
      begin
         NumberOfYears := PresentYear-PastYear;
         NumberOfMonths := PresentMonth+(NumberOfYears*12)+(12-trunc(MonthOfManufacture))
      end
      else
         NumberOfMonths := 0;

      Result := NumberOfMonths;
   except
    raise;
   end;
end;

function TBPFunctions.bp_CEILING(NumberToCeiling,
  Exponent: Double): Double;
var
  TempNum1 : extended;
  TempNum2 : Integer;
begin
  TempNum1 := NumberToCeiling * power(10, Exponent);
  TempNum2 := ceil(TempNum1);
  result := TempNum2 * power(10, (Exponent * (-1)));

end;

function TBPFunctions.bp_FLOOR(NumberToFloor, Exponent: Double): Double;
var
  TempNum1 : Extended;
  TempNum2 : Integer;
begin
  TempNum1 := NumberToFloor * power(10, Exponent);
  TempNum2 := Floor(TempNum1);
  result := TempNum2 * power(10, (Exponent * (-1)));

end;

function TBPFunctions.bp_THISYEAR: Double;
var
  Year, Month, Day : word;
begin
  DecodeDate(Date,Year,Month,Day);
  result := Year;
end;

initialization
  TAutoObjectFactory.Create(ComServer, TBPFunctions, Class_BPFunctions,
    ciMultiInstance, tmApartment);
end.
```

```
procedure TfrmBPMain.MinimizeDiscount;
var
  guess : extended;
  loop_count, ierrcode : word;
begin
  try
    try
      ValidateFields;

      loop_count := 0;
      guess := -999999999.00;
      Screen.Cursor:=crHourGlass;

      UpdateErrorMessage('Calculating.... Please Wait');

      //If deal structure not within possible parameters don't
      //bother to calculate discount
      ScreenToBPParameters;  // from screen to parameter
      Evaluate;
      ierrcode := ScriptControl.eval('ErrCode');
      BPParametersToScreen(false, false);                          //MODIFIED 21JUN01 JDP
      //If deal structure is within possible parameters then calculate discount
      if iErrCode <= 4 then
      begin
        edDiscount.asFloat := UNREASONABLE_DISCOUNT;
      end
      else begin
        UpdateErrorMessage('Minimizing Discount... Please Wait');
        edDiscount.asFloat := GetMinimumDiscount;
        while not WithInRange(guess) do
        begin
          if Loop_Count = MAX_LOOP_COUNT then
          begin
            break;
          end else
          begin
            guess := edDiscount.AsFloat;
            BuyProgramParameter.vRESERVE := edDiscount.AsFloat;
            Evaluate;
            edDiscount.AsFloat := GetNewDiscount;
            edDiscount.Update;   // updating the screen to show user the progress
            inc(loop_count);
          end;
        end;   // while

        BuyProgramParameter.vRESERVE := Math.MAX(edDiscount.AsFloat, guess);
        Evaluate;

        edDiscount.AsFloat := BuyProgramParameter.vRESERVE;
        if BuyProgramParameter.vStructOK = False then
        begin
          loop_count := 0;
          repeat
            edDiscount.AsFloat := edDiscount.AsFloat + JUMP_AMT;
            BuyProgramParameter.vRESERVE := edDiscount.AsFloat;
            Evaluate;
            edDiscount.AsFloat := BuyProgramParameter.vRESERVE;
            edDiscount.Update;  // update the screen
            if Loop_Count = MAX_LOOP_COUNT then
            begin
              UpdateDealMessage('Maximum Calculations Reached! Please recheck deal structure');
              break;
            end;
            Inc(Loop_Count);
          until BuyProgramParameter.vStructOK=True;
        end;
        Evaluate;
      end;
    except
      raise;
    end;
  finally
    Screen.Cursor:=crDefault;
    CalculateFlag := True;
    if BuyProgramParameter.vStructOK = True then
      UpdateErrorMessage('');
    BPParametersToScreen(false, false);                  //MODIFIED 21JUN01 JDP
  end;
end;
```

Explanation of User Input for Buy Program

Label on User Interface	Variable Name in Expressions	Meaning
Price	Price	Price of Vehicle
DOC	Doc	Documentary fee charged to customer
Smog	Smog	Fee for emissions testing and certificate
Sales Tax	TaxRate	Sales Tax Rate
Service Contract	Warr	Price of extended warranty sold to buyer
License	LicFee	Cost of License/Registration
Trade Allowance	TradeAllowance	Amount Dealer is giving for Trade-In
Trade Payoff	TradePayoff	Amount customer still owes on Trade-In
Cash Down	Down	Amount of Cash Down Payment
Insurance	Ins	(Y or N)=If customer will purchase Comp/Collision insurance with contract
Loan Date	Not Used	Date of contract
Date to 1st Pmt	DaysToPay	Number of days from contract date to First payment date
Payments	Term	Number of monthly payments in contract
Model Year	vYear	Year model of vehicle being sold
Blue Book	Book	Kelley Bluebook wholesale valuation Of vehicle being sold
Mileage	Miles	Miles on vehicle
Class	vClass	"Class" of vehicle per Westlake Guidelines
Cost	Cost	Dealer Cost of Vehicle/etc
Svc Cont Cost	WarCost	Dealer Cost of extended warranty

THE FOLLOWING VARIABLES ARE INPUT PER WESTLAKE GUIDELINES

Label on User Interface	Variable Name in Expressions
# Years on Credit Bureau	YrsTRW
# Good Credit Items	Good
$ High Good Credit	HiGood
# Derog Credit Items	Derog
$ High Derog Credit	HiDerog
# Of Repo/Auto Loss	Repos
Previous Bankruptcy?	BK
Customer Owns Home?	Home
Residence Stability	Resid
# Yrs On Job	Job
Gross Monthly Income	Inc
Rent/Mortgage	Rent
Family Support Debt	Support
Other Monthly Debt	Debt
Phone/Util/Chking in Name?	PhBill
Spouse/Partner Co-X?	Spouse
Other Co-X?	Cox
SP # Good	SpGood
SP $ High Good	SpHiGood
SP # Derog	SpDerog
SP High Derog	SpHiDerog
SP YrsJob	SpJob
SP Income	SpInc
COX # Good	CoxGood
COX # Derog	CoxDerog
COX # Repo	CoxRepo
COX Income	CoxInc
COX Owns Home	CoxHome
COX Parent	CoxParent

20501075.043501

United States Patent & Trademark Office

Office of Initial Patent Examination -- Scanning Division

10043579 .070902

Application deficiencies found during scanning:

☐ Page(s)__________of______________________________ were not present for scanning.
(Document title)

☐ Page(s)__________of______________________________ were not present for scanning.
(Document title)

DRAWINGS FIG 15 - 23 ARE DARK

☐ ***Scanned copy is best available.***